INRIA

CENTRE DE RENNES
IRISA

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél.:954 90 20

Rapports de Recherche

N° 115

RECURSION INDUCTION PRINCIPLE REVISITED

Gérard BOUDOL
Laurent KOTT

Janvier 1982

RÉCURSION INDUCTION
PRINCIPLE REVISITED

Gérard BOUDOL(1) and Laurent KOTT(2)

Publication n° 156 - Décembre 1981 - 49 pages

Abstract : Here we present a new version of recursion induction principle with an effective and, by the way, mechanisable flavour. Furthermore we obtain a measure of the complexity of equivalences (or inequalities) between recursive programs and also of the difficulty of their proofs.

Résumé : Nous présentons ici une nouvelle version du principe d'induction récurrent avec un souci d'effectivité et, donc, de mécanisation. Bien plus, nous obtenons une mesure de la complexité des équivalences (ou des inégalités) entre programmes récursifs et, aussi, de la difficulté de leurs preuves.

Authors' addresses (1) Université Paris VII
UER de Mathématiques L.I.T.P.
2, Place Jussieu
75251 PARIS CEDEX 05

(2) Université de Rennes I
UER de Mathématiques et Informatique
IRISA
Campus de Beaulieu
35042 RENNES CEDEX

Secrétariat de Rédaction : Melle Moinet F. - IRISA - Informatique
Université de Rennes, Campus de Beaulieu
35042 RENNES

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
(L.A. 227)
UNIVERSITÉ DE RENNES 1 I.N.S.A. DE RENNES

INSTITUT NATIONAL DE RECHERCHE
EN INFORMATIQUE ET EN AUTOMATIQUE
(LABORATOIRE DE RENNES)

I. Introduction.

Any manipulation, such as transformations, optimisation, development of programs, must be proved correct ; any verification process requires some methods, known to be valid, for proving properties of programs.

For this purpose, in the years 60, the litterature presents two kinds of very useful tools for the recursive case :

- a principle of structural induction (R.M. Burstall [5])
- a principle of recursion induction (J.W. de Baker, J.Mc Carthy, J.H. Morris, D. Park and D. Scott [3, 25, 27, 30])

or, more precisely several statements of recursion induction principle which are equivalent roughly speaking (see I. Greif [14]).

Our aim is to study proofs performed in a formal system which manipulates universally quantified first order formulas, by rewriting systems on terms (close to some methods presented by G. Huet and D. Oppen [21]). Thus we deal only with recursion induction principle, for the structural induction requires a very deep knowledge about calculi domains which is not in general finitely axiomatisable by first-order theory. The origin of this restriction lies in the research of a notion of "effectiveness of induction principle" in order to design a system, more or less mechanisable, performing induction proofs. Indeed the methods developped in this framework give some ability to understand why certain proofs require human skilfulness (namely invention of auxiliary lemmas). Indeed we construct a proofs system such that we are able to associate in a constructive way a recursive function over integers to each demonstration performed : this function gives a measure of the difficulty of this demonstration. Moreover if we deal with * known as difficult (a priori an intuitive notion but which is defined precisely below) and provable with the system then the demonstration performed is difficult necessaraly. To give some intuition about effectiveness, quickly recall some definitions and results of the well known fixed point theory.

Let D be a domain (partially) ordered by $\sqsubseteq$ and with a least element $\perp$ such that (D, $\sqsubseteq$, $\perp$) is a c.p.o. ; note $(D^n \to D)$ the set of continuous function with n arguments over D and call too $\sqsubseteq$ the canonical extension of the order relation on D. A recursive program is an equation (actually a system of equations) like

$$F(\vec{X}) = \tau[F](\vec{X})$$

* proofs

where $\vec{X}$ is a n-vector of variables and τ a continuous functionnal over $(D^n \to D)$.

By definition the function computed by this program is the least fixed point of the functionnal τ, called Y_τ, which is the least upper bound of the increasing sequence of elements of $(D^n \to D)$ $\{\tau^n[\perp] \mid n \in N\}$ where $\perp$ denotes too the constant function equal to $\perp$.

Let P be a predicate and τ be a continuous functionnal, both over $(D^n \to D)$.

The recursive induction principle states

if (i) $P(\perp)$

and (ii) $\forall f \in (D^n \to D)\ (P(f) \Rightarrow P(\tau[f]))$

are true then Y_τ satisfies P (obviously P cannot be any predicate and must belong to the class of admissible predicates, see Z. Manna [26]).

It is clear that this statement is too general and may be written with a more effective flavour. For example

if (i) $\exists p \in N\ P(\tau^p[\perp])$

and (ii) $\forall n \in N\ (\forall k \in N(p \leq k \leq n \wedge P(\tau^k[\perp]) \Rightarrow P(\tau^{n+1}[\perp])$

are true then Y_τ satisfies P.

Let us see an example introducing notations used below.

Example 1 : Let D be a c.p.o. and R be the system of equations

$F(x) = f(F(x))$ $\quad\tau$ functionnal over $(D \to D)$ $\lambda F.fF$

$G(x) = g(G(x))$ $\quad\sigma$ functionnal over $(D \to D)$ $\lambda G.gG$

Assume that f and g check the set of properties S $\{f(g(x)) = g(f(x)),\ f(\perp) = g(\perp)\}$. We are going to show that Y_τ and Y_σ are equal by recursion induction stated above :

(i) we choose p equal to 1, because $\tau[\perp]$ is equal to $f(\perp)$ and $\sigma[\perp]$ to $g(\perp)$ and by the second property of S (i) is true ;

(ii) the induction hypothesis is : for each $n \geq 1$ $\tau^n[\perp]$ is equivalent to $\sigma^n[\perp]$. Then

$$
\begin{aligned}
\tau^{n+1}[\perp] &= f(\tau^n[\perp]) && \text{by definition of } \tau \\
&= f(\sigma^n[\perp]) && \text{by induction hypothesis} \\
&= f(g(\sigma^{n-1}[\perp])) && \text{by definition of } \sigma \\
&= g(f(\sigma^{n-1}[\perp])) && \text{by the } 1^{st} \text{ property of } S \\
&= g(f(\tau^{n-1}[\perp])) && \text{by induction hypothesis} \\
&= g(\tau^n[\perp]) && \text{by definition of } \tau \\
&= g(\sigma^n[\perp]) && \text{by induction hypothesis} \\
&= \sigma^{n+1}[\perp] && \text{by definition of } \sigma
\end{aligned}
$$

and we reach the equality $\tau^{n+1}[\perp] = \sigma^{n+1}[\perp]$. So (ii) is true too and the principle ensures us the equivalence between Y_τ and Y_σ. We remark that we start from the term $\tau^{n+1}[\perp]$ and reach the term $\sigma^{n+1}[\perp]$ by a sequence of terms rewriting following three rules :

- equations of the recursive program definition R
- laws satisfied by base functions S
- induction hypothesis.

Our point of view about induction principle and system of equivalence proofs involves some constraints which are adequatly formalised in the framework of the theory of algebraic semantics [10, 11, 15, 16, 28, 29] :

- recursive programs (schemes, actually) are systems of equations between well formed terms built from a set A of base function symbols, a set F of procedures symbols and a set V of variables ;

- laws (properties) of base function symbols are expressed as axiomatic system, subset S of the cartesian product $M_A(V) \times M_A(V)$ (where $M_A(V)$ is the set of basic terms), and are used in proofs by mean of the generated congruence $\overset{*}{\underset{S}{\longleftrightarrow}}$ (i.e. the rewriting relation defined by S, see below) ;

- induction proofs are induction on the Kleene's sequences ; the Kleene's sequence related to some functionnal is the set $\{\tau^n[\perp] \mid n \in N\}$ where $\tau^n[\perp]$ is obtained by n consecutive calls of the full substitution rule on $\tau[\perp]$;

- formulas are (conjonction of) inequalities between terms ; we shall write (τ,σ) instead of $\tau \sqsubseteq \sigma$.

We can state a restricted form of the classical induction principle (Morris [27]) as follows

if for any integer n $\tau^n[\perp](\underset{S}{\longleftrightarrow} \cup \underset{I_n}{\longrightarrow} \cup \underset{R}{\longleftrightarrow})^* \sigma^n[\perp]$
then $\tau \leq_{<R,S>} \sigma$.

where - $\leq_{<R,S>}$ means that for every interpretation satisfying S the function computed by τ with the system of recursive definitions R is less defined than the function computed by σ ;

- $\underset{S}{\longrightarrow}$ is the rewritting relation generated by S ;
- $\underset{I_n}{\longrightarrow}$ is the rewritting relation generated by the set $I_n = \{(\tau^p[\perp], \sigma^p[\perp]) \mid p<n\}$
- $\underset{R}{\longleftrightarrow}$ is $\underset{R}{\longrightarrow} \cup \underset{R'}{\longrightarrow}$
- $(\underset{S}{\longleftrightarrow} \cup \underset{I_n}{\longleftrightarrow} \cup \underset{R}{\longleftrightarrow})^*$ is the reflexive and transitive closure of $\underset{S}{\longleftrightarrow} \cup \underset{I_n}{\longrightarrow} \cup \underset{R}{\longleftrightarrow}$.

Here we get an explicit induction step by means of the relation $\underset{I_n}{\longrightarrow}$, and we can rewrite the example 1 with these notations as the reader could check. However some examples show that this statement is too restrictive.

<u>Example 2</u> : Let us consider the system of recursive definitions R

$$\begin{cases} F(x) = h(F(x)) & (\tau = \lambda F.hF) \\ G(x) = h(h(G(x))) & (\sigma = \lambda G.hhG) \end{cases}$$

One easily checks

(i) $\sigma^0[\perp] = \tau^0[\perp]$

(ii) $\forall j \in N \;\; \sigma^j[\perp] = \tau^{2j}[\perp] \Rightarrow \sigma^{j+1}[\perp] = \tau^{2(j+1)}[\perp]$

which ensures us the identity $Y_\tau = Y_\sigma$.

<u>Example 3</u> : (R. Milner [2]) Let R be the system of recursive definitions

$$\begin{cases} F(x) = f(F(x)) & (\tau = \lambda F.fF) \\ G(x) = g(G(x)) & (\sigma = \lambda G.gG) \end{cases}$$

and S be the set of laws $\{f(g(x)) = g^2(f(x)),\ f(\perp) = g(\perp)\}$, we have the following properties (easily provable)

(i) $\sigma[\perp] \equiv_{<R,S>} \tau[\perp]$

(ii) $\forall j \in N \;\; f(\sigma^j[\perp]) \equiv_{<R,S>} \sigma^{2j+1}[\perp]$

(iii) $\forall j \in N \;\; \tau^j[\perp] \equiv_{<R,S>} \sigma^{2^j-1}[\perp]$

then $\tau \equiv_{<R,S>} \sigma$ i.e. τ and σ have the same least fixed point for any interpretation of f and g satisfying the set of laws S.

We have to generalise our induction principle by choosing for each pair of terms (τ,σ) a suitable sequence of pairs of approximations of the functions computed from σ and τ by the system R of recursive definitions instead of the Kleene's sequence. But to keep some gain in effectivity we consider subsequences - selected by a function k from N to N - of the sequence of approximations obtained by applying some universally correct computation rule ρ [7, 13, 34] (the most often the "full substitution" or the "parallel outermost" rule). Then we restate our principle

if for any integer n $\rho^{n}(\tau)[\perp] \; (\underset{S}{\longleftrightarrow} \cup \underset{I_n^k}{\longrightarrow} \cup \underset{R}{\longleftrightarrow})^{*} \; \rho^{k(n)}(\sigma)[\perp]$

then $\tau \leqslant_{<R,S>} \sigma$.

where $\underset{I_n^k}{\longrightarrow}$ is the rewriting relation generated by the set

$$I_n^k = \{(\rho^{p}(\tau)[\perp], \rho^{k(p)}(\sigma)[\perp]) \mid p<n\}$$

This new formulation (well fitted to the former examples) suggests the main concept of this article, namely classes of formulas related to (universally) correct computation rule and some class of functions from N to N : we shall note $I_R^S(\rho,\mathfrak{F})$ the set of formulas, that is couples of terms (τ,σ), such that there exists k of $\mathfrak{F}$ such that, for any integer n $\rho^{n}(\tau)[\perp] \sqsubseteq_S \rho^{k(n)}(\sigma)[\perp]$ (where $t \sqsubseteq_S t'$ means the interpretation of t is less defined than that of t' for any interpretation satisfying S) and write $E_R^S(\rho,\mathfrak{F})$ $\{(\tau,\sigma) \mid (\tau,\sigma) \in I_R^S(\rho,\mathfrak{F})$ and $(\sigma,\tau) \in I_R^S(\rho,\mathfrak{F})\}$. Clearly for any formula (τ,σ) in $I_R^S(\rho,\mathfrak{F})$ the inequality $\tau \leqslant_{<R,S>} \sigma$ is valid, and the same validity result holds for $E_R^S(\rho,\mathfrak{F})$:

$$(\tau,\sigma) \in E_R^S(\rho,\mathfrak{F}) \Rightarrow \tau \equiv_{<R,S>} \sigma$$

We can see the fact that a formula (τ,σ) belongs to some $I_R^S(\rho,\mathfrak{F})$ as an indication (a measure, by means of the class of functions $\mathfrak{F}$) of the difficulty or complexity of a proof of the theorem $\tau \leqslant_{<R,S>} \sigma$ (indeed in the examples one can see that a "difficult" theorem in this sense requires the help of some auxiliary lemmas). On the other hand, the fact $(\tau,\sigma) \in E_R^S(\rho,\mathfrak{F})$ gives some idea about the relative "efficiency of computation" of the same function by τ and σ, with respect to the computation rule ρ.

From this point of view of "proofs complexity" we investigate the problem of completeness of these classes of formulas and obtain the following results where we denote respectively fs and po the "full substitution" and "parallel outermost" (or "parallel call by name") computation rules (see [3, 13, 34]) :

(0) For any recursive function f from N to N, there exists a recursive program R, a set of axioms S and a formula such that $\tau \leqslant_{<R,S>} \sigma$, and $(\tau,\sigma) \in I_R^S(fs,\{g\})$ and $(\tau,\sigma) \in I_R^S(po,\{g\})$ imply $g \geqslant f$ (that is $\forall n \in N$ $g(n) \geqslant f(n)$).

(1) For any recursive program R and formula (τ,σ) such that $\tau \leqslant_R \sigma$ (strong or syntactic inequality) then (τ,σ) belongs to $I_R(fs,Exp)$ where Exp is the set of exponential functions form N to N. Moreover there exists R and (τ,σ) such that $\tau \leqslant_R \sigma$ and $(\tau,\sigma) \in I_R(fs,\{g\})$ imply $\forall n \in N$ $g(n) \geqslant \frac{n^2+n}{2}$.

(2) For any R and (τ,σ) such that $\tau \leqslant_R \sigma$, the formula (τ,σ) belongs to $I_R(po, Lin)$ where Lin is the set of linear functions, a result which explains the better suitability of "po-induction" (see [4]) for proving stong equivalences.

(3) If R is a "non-nested" (or linear) system of recursive definitions and $\tau \equiv_R \sigma$, then (τ,σ) belongs to $E_R(fs,Lin)$ (and in this case $\equiv_R$ is decidable).

These three results are obtained by using the language of branches of the tree generated by a term in a recursive program.

(4) May be the most interesting and main result of this article is obtained in the study of formal system of proofs : we define a powerful system to prove formulas, which features follow these designed by B. Courcelle and J. Vuillemin [12], namely fs-induction and ability to consider procedure symbols as least fixed points by duplicating the system of recursive definitions. We carefully study the condition of application of the fs-induction inference rule and show that if a formula (τ,σ) is provable in this system, with R (recursive program) and S (axioms) as hypothesis, then (τ,σ) belongs to $I_R^S(fs,\mathfrak{F})$ where $\mathfrak{F}$ is the set of recursive functions, proving thus the validity of the system (a proof which is not always given in the related litterature). Furthermore, given a proof of a formula (τ,σ), we are able to construct a function f of $\mathfrak{F}$ such that (τ,σ) belongs to $I_R^S(fs,\{f\})$.

Moreover if we do not allow duplication of recursive programs then a provable formula $\tau \leq_{<R,S>} \sigma$ belongs to $I_R^S(fs,Lin)$.

We believe these results and methods raise very interesting problems and open a wide research area. In particular in the field of methods for proving program properties [9, 19, 20, 24], it seems interesting to study their power by means of this notion of complexity ; for example it has been proved in [23] that the so-called fold/unfold method of R. Burstall and J. Darlington [6] allows to prove po-linear equivalences only.

II. Algebraic framework.

In the following sections we briefly recall formalism and main results of the algebraic semantics of recursive program schemes. For more details we urge the reader to refer to [1, 10, 11, 16, 28, 29].

Notations : we shall use some notations

N is the set of positive intergers ;

for any k of N greater then 0 $[k]$ denotes the set $\{i \in N \mid 1 \leq i \leq k\}$,

$[0]$ may denote the empty set ;

for any non empty set X the free monoïd generated by X is X^*, the empty word Λ, and the integer $|u|$ is the length of the word u ;

for any subset L of X^* and any word w in X^* $L/w = \{u \mid u \in X^* \text{ and } w.u \in L\}$.

§1 C.p.o. set

Definition 1 - $(D, \sqsubseteq, \perp)$ is a complete partially order (abbreviated in c.p.o.) if and only if $\perp$ is the least element of the partially ordered set $(D, \sqsubseteq)$ and each directed subset Δ of D has a least upper bound denoted $U\Delta$.

Let us recall that Δ is a directed subset of D if and only if for any pair (d,d') of elements of Δ there exists d'' of Δ such that $d \sqsubseteq d'$ and $d' \sqsubseteq d''$.

Remark - Let be $(D,\sqsubseteq)$ a partially ordered set, the set D^k (k in N, $k>0$) will be always ordered component wise and we shall use the same notation

Definition 2 - Let $(D,\sqsubseteq)$ and $(D', \sqsubseteq')$ be partially ordered sets. A mapping f from D^k to D' is increasing iff $\forall (d_1,\dots,d_k)$, $\forall (e_1,\dots,e_k) \in D^k$. $(d_1,\dots,d_k) \sqsubseteq (e_1,\dots,e_k) \Rightarrow f(d_1,\dots,d_k)$ $f(e_1,\dots,e_k)$.

Furthermore f is continuous iff for any directed subsets of D, $\Delta_1,\dots,\Delta_k$, having least upper bound, then the set $f(\Delta_1,\dots,\Delta_k)$ is directed (with respect to $\sqsubseteq'$) and admits $f(U\Delta_1,\dots,U\Delta_k)$ as least upper bound.

§2 F-magma

In order to built recursive program schemes which are systems of equations on terms, we need a set of function symbols, say F, whose elements are symbols with arity (a non negative integer). We call F_k the set of elements of F with the same arity k.

Well formed terms (with respect to arities) are obtained by composition of these symbols, applied to variables, and may be viewed as particular cases of finite trees and the set of trees -finite or infinite- is a special case of F-magma (or F-algebra).

Definition 1 - An ordered F-magma is a structure $M = < D_M, \sqsubseteq_M, \perp_M, \{f_M \mid f \in F\} >$ where $(D_M, \sqsubseteq_M)$ is a partial ordered set with $\perp_M$ as least element and, for any f of F_k, f_M is an increasing mapping from D_M^k to D_M. M is said complete if $(D_M, \sqsubseteq_M, \perp_M)$ is a c.p.o. and each f_M is continuous.

A morphism h between two complete ordered F-magmas, M and M', is a continuous mapping from D_M to $D_{M'}$ which preserves their structure.

Definition 2 - A complete ordered F-magma (abbreviated in coF-M) M is free over X if and only if X is included in D_M (up to a canonical injection) and, for any other coF-M M' and any mapping h from X to $D_{M'}$, there exists a unique morphism h_M^∞, from M to M' whose the restriction to X is identical to h.

Obviously, such a magma is defined up to an isomorphism and we are going to show the free coF-M over X is isomorphic to the set of F-well formed trees over X.

§3 Trees

Let be V a set of variables disjoint from F and described by $\{x_n \mid n \in N\}$. Let us introduce the splitted alphabet associated to F (see [8])
$W_F = \{(f,i) \mid f \in F_k, i \in N\}$.

Definition 1 - A F-well formed tree on V (abbreviated in a F-tree on V) is amapping t from dom(t), a subset of W_F^*, to F U V such that $\forall w \in W_F^*, \forall f \in F_k$ (k>0) $\forall (f,i) \in W_F$: $w.(f,i) \in dom(t)$ implies :

(i) $w \in dom(t)$

(ii) $t(w) = f$

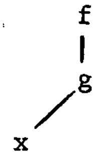
(iii) $W_F \cap dom(t)/w \subseteq \{(f,j) \mid j \in [k]\}$

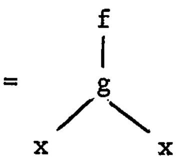
Example 1 - $V = \{x\}$, $F = F_1 \cup F_2$, $F_1 = \{f\}$, $F_2 = \{g\}$. t_1 is the application s.t. $dom(t_1) = \{\Lambda, (f,1), (f,1)(g,1)\}$; $t_1(\Lambda) = f$; $t_1((f,1)) = g$; $t_1((f,1)(g,1) = x$. t_2 is the application s.t. $dom(t_2) = \{\Lambda, (f,1), (f,1)(g,1), (f,1),(g,2)\}$; $t_2(\Lambda) = f$; $t_2((f,1)) = g$; $t_2((f,1)(g,1)) = t_2((f,1)(g,2)) = x$.

t_3 is the application s.t. $\mathrm{dom}(t_3) = \{\Lambda,(f,1),(f,2),(f,1)(g,1),(f,1)(g,2)\}$; $t_3(\Lambda) = f$; $t_3((f,1)) = g$; $t_3((f,2)) = x$; $t_3((f,1)) = t_3((f,1)(g,2) = x$.

Both t_1 and t_2 satisfy the definition 1 above but t_3 does not because f has an arity equal to 1 and $W_F \cap \mathrm{dom}(t_3)/\Lambda = \{(f,1),(f,2)\}$.

As usual t_1 and t_2 may be drawn as follows

$t_1 =$ 

$t_2 =$ 

(in some intuitive sense our trees are not complete with respect to arities).

We note $M_F^\infty(V)$ the set of F-trees on V and we define on this set the syntactic order by $\forall t \in M_F^\infty(V)$, $\forall t' \in M_F^\infty(V)$ $t \sqsubseteq t'$ iff $\mathrm{dom}(t) \subseteq \mathrm{dom}(t')$ and, for any w in dom(t), t(w) is equal to t'(w).

In the example 1 above t_1 is less than t_2. For this order the empty tree Ω (whom the domain is empty) is the least element of $M_F^\infty(V)$.

<u>Proposition</u> - The structure $H =< M_F(V), \sqsubseteq, \Omega, \{f_H \mid f \in F\}$ is the free coF-M over V ; where, for any f of F_k, f_H is an application from $M_F^\infty(V)^k$ to $M_F^\infty(V)$ s.t., for any $(t_1,\ldots,t_k)$ of $M_F^\infty(V)^k$, $f_H(t_1,\ldots,t_k)$ is a tree t defined by
$\mathrm{dom}(t) = \{\Lambda\} \cup \bigsqcup_{i\in[k]} (f,i).\mathrm{dom}(t_i)$; $t(\Lambda) = f$; $t((f,i).w) = t_i(w)$ for any i in [k] and w in $\mathrm{dom}(t_i)$.

To achieve this result it suffices to check (and checking is rather tedious) the conditions of the definition 2 of §2.

If we note $M_F(V)$ the set of finite F-trees(whom the domain is finite) and $\overline{M}_F(V)$ the set of finite F-trees which are $\sqsubseteq$-maximal (represented by terms denoted as usually by prefixed polish notation), then $<\overline{M}_F(V)$, $\{f_H \mid f \in F\}>$ is the free F-magma on V and $<M_F(V), \sqsubseteq, \Omega, \{f_H \mid f \in F\}>$ is the free ordered F-magma on V (with respect of an obvious definition of these structures).

For any mapping ν from V to $M_F^\infty(V)$, we note ν^* (instead of ν_H^∞ as written in def. 2 of §2) the unique endomorphism of H whom the restriction to V is ν. We call it a <u>substitution</u>. Sometimes we write $t[t_1/x_{i_1},\ldots,t_p/x_{i_p}]$ instead of ν (t) where $\{x_{i_j} \mid j \in [p]\}$ is the set of elements of V occuring at least once in t and, for any j in [p], $\nu(x_{i_j}) = t_j$.

§4 Congruences on H

We shall use the concept of congruences on F-magma and, more precisely, congruences on H stable by substitution (see [18, 31]). To define this restricted notion we need the concept of subtree and subtree replacement. First we note dom(t), the enlarged domain of the tree t in $M_F^\infty(V)$, the set $\{w.(f,i)/w \in dom(t),\ t(w) = f,\ f \in F_k,\ i \in [k]\}$ if $t \neq \Omega$ and $dom(\Omega) = \{\Lambda\}$.

Definition 1 - For any w in dom(t), the subtree of t at node w, denoted t/w, is defined by dom(t/w) = dom(t)/w and, for any u in dom(t/w), t/w(u) = t(w.u). Furthermore, for any t' in $M_F^\infty(V)$, the tree t[t'/w] obtained by the replacement of t/w by t' in t is defined by $dom(t[t'/w]) = (dom(t)-w.W_F^*) \cup w.dom(t')$, for any u in $dom(t)-w.W_F^*$, $t[t'/w](u)$, for any u in w. dom(t') t[t'/w](w.u) = t'(u).

Definition 2 - A precongruence on $M_F^\infty(V)$ is a preorder (a reflexive and transitive relation) R on $M_F^\infty(V) x M_F^\infty(V)$ such that, for any t in $M_F^\infty(V)$, any w in dom(t), any mapping ν from V to $M_F^\infty(V)$ and any pair (s,s') of trees, (s,s') in R implies the pair $(t[\nu^*(s)/w], t[\nu^*(s')/w])$ belongs to R too. A congruence on $M_F^\infty(V)$ is a symetric precongruence.

Proposition - For any R, viewed as a subset of the cartesian product $M_F^\infty(V) x M_F^\infty(V)$, the precongruence generated by R (i.e. the least precongruence including R) is the reflexive and transitive closure of the relation $\xrightarrow[R]{}$ defined by $t \xrightarrow[R]{} t'$ iff

$\exists (s,s') \in R$, $\exists w \in dom(t)$, $\exists \nu : V \longrightarrow M_F^\infty(V)$ s.t.
$t = t[\nu^*(s)/w]$ and $t' = t[\nu^*(s')/w]$ while the congruence generated by R is the reflexive and transitive closure of the relation $\xrightarrow[R]{} \cup \xrightarrow[R^{-1}]{}$.

As usual we shall note $\xrightarrow[R]{*}$ (resp. $\xleftrightarrow[R]{*}$) the precongruence (resp. congruence) generated by R.

Example - $V = \{x\}$; $F = F_1 = \{f,g\}$; $R = \{(f\Omega = g\Omega), (fgx = ggfx)\}$. Considering the precongruence generated by R, it is easy to show by integer induction that $f^n\Omega \xrightarrow[R]{*} g^{2^n-1}\Omega$.

III. Recursive program schemes and their semantics.

§ 1 Recursive program schemes

A Recursive Program Scheme (RPS) is a triple $\Sigma = (A, \Phi, R)$ where

- A is the base function symbols' alphabet and is some finite subset of $\bigsqcup_{n\in N} A_n$; $A_n = \{a_p^n \mid p \in N$ and the arity of a_p^n is $n\}$;

- Φ is the procedure symbols' alphabet and is equal to $\{\varphi_1,\ldots,\varphi_N\}$; each φ_i has an arity equal to n_i ;

- R is a functionnal binary relation over $\overline{M}_{A\cup\Phi}(V)$ such that if (s,t) belongs to R there exists an integer i (in [N]) such that s is equal to $\varphi_i x_1\ldots x_{n_i}$ and t belongs to $\overline{M}_{A\cup\Phi}(\{x_1,\ldots,x_{n_i}\})$.

For sake of clarity, we shall write $\xrightarrow[\Sigma]{}$ instead of $\xrightarrow[R]{}$ and use the alternative presentation for RPSs

$$\Sigma\begin{cases}\varphi_i x_1\ldots x_{n_i} = \tau_i, \ \tau_i \in \overline{M}_{A\cup\Phi}(\{x_1,\ldots,x_{n_i}\})\\ 1 \leqslant i \leqslant N\end{cases}$$

Example - $A = A_2 = \{a\}$; $\Phi = \Phi_1 = \{\varphi_1, \varphi_2, \varphi_3\}$; $V = \{x\}$; let us consider the following RPS

$$\begin{cases}\varphi_1 x = ax\varphi_2\varphi_1 x\\ \varphi_2 x = a\varphi_1 x\varphi_2\varphi_2 x\\ \varphi_3 x = ax\varphi_3\varphi_3\varphi_3 x\end{cases}$$

Computations of a term s in a RPS, Σ, (A, Φ, R), are needed to define the semantics (see below) and are sequences ot terms rewrited from s in Σ. The direct or immediate information contained in a term s is all what we can know about this term without making any computation, that is to say by ignoring the value of procedure symbols occurring in s. Whence the definition of immediate information $\pi(s)$ of a term s :

$\pi(x) = x$ for any x in V

$\pi(a_p^n t_1\ldots t_p) = a_p^n \pi(t_1)\ldots\pi(t_p)$ for a_p^n in A

$\pi(\varphi_i t_1\ldots t_{n_i}) = \Omega$

thus π is a mapping from $\overline{M}_{A\cup\Phi}(V)$ to $M_A(V)$.

Computation may be defined more precisely by sequence of iterated application of some computation rule. Here we can define the fs and po computation rules (related to a RPS Σ) mentionned in the introduction as mapping σ_Σ and α_Σ on $\overline{M}_{AU}$ (V) respectively by

$$\sigma_\Sigma(x) = \alpha_\Sigma(x) = x \text{ for any } x \text{ in } V$$

$$\sigma_\Sigma(a_p^n t_1 \ldots t_p) = a_p^n \sigma_\Sigma(t_1) \ldots \sigma_\Sigma(t_p) \text{ and}$$

$$\alpha_\Sigma(a_p^n t_1 \ldots t_p) = a_p^n \alpha_\Sigma(t_1) \ldots \alpha_\Sigma(t_p) \text{ for any } a_p^n \text{ in } A$$

$$\sigma_\Sigma(\varphi_i t_1 \ldots t_{n_i}) = \tau_i[\sigma_\Sigma(t_1)/x_1 \ldots \sigma_\Sigma(t_{n_i})/x_{n_i}] \text{ and}$$

$$\alpha_\Sigma(\varphi_i t_1 \ldots t_{n_i}) = \tau_i[t_1/x_1 \ldots t_{n_i}/x_{n_i}] \text{ for any } \varphi_i \text{ in } \Phi.$$

Example 1 - (cont.)

$$\alpha_\Sigma(\varphi_1 \varphi_1 x) = a\varphi_1 x \varphi_2 \varphi_1 \varphi_1 x$$

$$\sigma_\Sigma(\varphi_1 \varphi_1 x) = aax\varphi_2 \varphi_1 x \varphi_2 \varphi_1 ax \varphi_2 \varphi_1 x$$

We leave to the reader to check the following facts :

Let Σ be a RPS, s and s' be elements of $\overline{M}_{AU\Phi}(V)$

(1) $s \sqsubseteq s' \Rightarrow \pi(s) \sqsubseteq \pi(s')$ and $\sigma_\Sigma(s) \sqsubseteq \sigma_\Sigma(s')$

(2) if there exists p terms of $\overline{M}_{AU\Phi}(V)$, $t_1,\ldots,t_p$, and p substitutions $\nu_1,\ldots,\nu_p$, such that $s = s'[\nu_1^*(t_1)/x_{i_1},\ldots,\nu_p^*(t_p)/x_{i_p}]$ then

$$\pi(s) = \pi(s')[(\pi o \nu_1)^*(\pi(t_1))/x_{i_1},\ldots,(\pi o \nu_p)^*(\pi(t_p))/x_{i_p}]$$

$$\sigma_\Sigma(s) = \sigma_\Sigma(s')[(\sigma_\Sigma o \nu_1)^*(\sigma_\Sigma(t_1))/x_{i_1},\ldots,(\sigma_\Sigma o \nu p)^*(\sigma_\Sigma(t_p))/x_{i_p}]$$

§2 Semantics of RPSs

Let Σ, (A, Φ, R), be a RPS, an interpretation of Σ is a complete ordered A-magma $M = \langle D_M, \leq_M, \perp_M, \{a_M \mid a \in A\}\rangle$. We call D_M^V the set of "data mappings" naturally ordered by the relation, also denoted $\sqsubseteq_M$, defined by, for some ν, ν', in D_M^V, $\nu \sqsubseteq_M \nu'$ iff $\nu(x) \sqsubseteq_M \nu'(x)$ for any x in V.

Since $M_A^\infty(V)$ is the free complete ordered A-magma over V, for each t belonging to $M_A^\infty(V)$ and each interpretation M, we are able to define a continuous function t_M from D_M^V to D_M by $t_M(\nu)$ equals to $\nu_M^\infty(t)$ for ν in D_M^V.

Theorem 1 - (M. Nivat [28])

For any RPS $\Sigma(=(A, \Phi, R))$, any s of $\overline{M}_{A\cup\Phi}(V)$ and any interpretation M of Σ the set $\{\pi(t)_M \mid s \xrightarrow[\Sigma]{*} t\}$ is directed with respect to $\sqsubseteq_M$.

Now we define the function computed by any term s of $\overline{M}_{A\cup\Phi}(V)$ for a given RPS Σ and an interpretation M.

Definition 1 - Let Σ, (A, Φ, R), M be an interpretation of Σ and s be a term of $\overline{M}_{A\cup\Phi}(V)$, the function computed by s is noted $s_{<\Sigma,M>}$ and is the least upperbound of the set $\{\pi(t)_M \mid s \xrightarrow[\Sigma]{*} t\}$.

When M is the free interpretation H ($<M_A^\infty(V), \sqsubseteq, \Omega, \{a_H \mid a \in A\}>$ cf. II.§3) we write s_Σ instead of $s_{<\Sigma,H>}$ and we have the following proposition.

Proposition 1 - Let Σ, (A, Φ, R) be a RPS, M be an interpretation and s be a term of $\overline{M}_{A\cup\Phi}(V)$, actually s_Σ is the tree generated by s in Σ and belongs to $M_A^\infty(V)$, so we can define the function $(s_\Sigma)_M$ and we have the identity

$$(s_\Sigma)_M = s_{<\Sigma,M>}$$

The following proposition rely computed function and rule of computation.

Proposition 2 - Let Σ, (A, Φ, R) be a RPS, M be an interpretation and s be a term of $\overline{M}_{A\cup\Phi}(V)$, $s_{<\Sigma,M>}$ is the least upperbound of $\{(\pi\alpha^n(s))_M \mid n \in N\}$ and of $\{(\pi\sigma^n(s))_M \mid n \in N\}$ too.

We could generalize this result by introducing the notion of correct computation rule ; let us recall that a computation rule ρ associates to each RPS Σ, (A, Φ, R), an application from $\overline{M}_{A\cup\Phi}(V)$ to itself such that, for any s in $\overline{M}_{A\cup\Phi}(V)$, $s \xrightarrow[\Sigma]{*} \rho_\Sigma(s)$; ρ is said correct if and only if, for any s of $\overline{M}_{A\cup\Phi}(V)$ and any interpretation M, the set $\{(\pi\rho_\Sigma^n(s))_M \mid n \in N\}$ admits $s_{<\Sigma,M>}$ as least upperbound. We deduce from the above proposition that α and σ are correct computation rules [7, 13, 34].

§3 Equivalence and inequality modulo a class of interpretations

One of the main advantages of the algebraic semantics is related to the equivalence of RPSs ; so we introduce a preorder and an equivalence over $\overline{M}_{A\cup\Phi}(V)$.

<u>Definition 1</u> - Let Σ, (A, Φ, R), a RPS and C be a class of interpretations, we define the preorder relation $\leqslant_{<\Sigma,C>}$ over $\overline{M}_{A\cup\Phi}(V)$ by $\forall s, s' \in \overline{M}_{A\cup\Phi}(V)\ s \leqslant_{<\Sigma,C>} s' \Leftrightarrow \forall M \in C\ s_{<\Sigma,M>} \leqslant_M s'_{<\Sigma,M>}$ and we note $\equiv_{<\Sigma,M>}$ the associated equivalence relation.

A class C of interpretations is said <u>relationnal</u> if the following holds : there exists a binary relation over $M_A(V)$ such that for any interpretation M, $M \in C \Leftrightarrow ((s,t) \in S \Rightarrow s_M =_M t_M)$.

For a given relation S, we shall note C_S the relationnal class associated to S and write $\leqslant_{<\Sigma,S>}$ (resp. $\equiv_{<\Sigma,S>}$) instead of $\leqslant_{<\Sigma,C_S>}$ (resp. $\equiv_{<\Sigma,C_S>}$). Now we give a very fruitful theorem which generalizes the theorem 1 of M. Nivat and will be used very often in the sequel.

<u>Theorem 2</u> - (I. Guessarian [16])

Let Σ, (A, Φ, R), be a RPS, (s,s') be a pair of terms belonging to $\overline{M}_{A\cup\Phi}(V)$ and S a subset of $M_A(M) x M_A(V)$; then

$$s \leqslant_{\Sigma,S>} s' \Leftrightarrow \forall t : s \xrightarrow[\Sigma]{*} t\ \exists t' : s' \xrightarrow[\Sigma]{*} t' \text{ s.t. } \pi(t) \sqsubseteq_S \pi(t')$$

where $\sqsubseteq_S$ is the precongruence $(\sqsubseteq \cup \xleftrightarrow[S]{})^*$ over $M_A(V)$.

If $\mathcal{A}$ is the class of all interpretations we shall write $\leqslant_\Sigma$ (resp. $\equiv_\Sigma$) instead of $\leqslant_{<\Sigma,\mathcal{A}>}$ (resp. $\equiv_{<\Sigma,\mathcal{A}>}$).

<u>Proposition 1</u>-[29]Let Σ,(A, Φ, R), be a RPS and (s,s') be a pair of elements in $\overline{M}_{A\cup\Phi}(V)$; then the following property holds

$$s \leqslant_\Sigma s' \text{ if and only is } s_\Sigma \sqsubseteq s'_\Sigma$$

<u>Proof</u> : <u>if part</u> H is a particular interpretation, then by definition of $\leqslant_\Sigma$, the inequality $s \leqslant_\Sigma s'$ implies $s_{<\Sigma,H>} \sqsubseteq_H s'_{<\Sigma,H>}$. Following our notation the last inequality may be rewritten $s_\Sigma \sqsubseteq s'_\Sigma$.

<u>only if part</u> For any interpretation M we deduce from the inequality $s_\Sigma \sqsubseteq s'_\Sigma$ the inequality $(s_\Sigma)_M \sqsubseteq_M (s'_\Sigma)_M$ and, by the proposition 2 of §2, we reach the inequality $s_{<\Sigma,M>} \sqsubseteq_M s'_{<\Sigma,M>}$ which holds for any interpretation M. So we obtain the result.

§4 <u>Representation of trees by languages</u>

In this paragraph we define the representation of trees by their languages of branches, following an idea due to W. Rounds [32] and used in semantics by B. Rosen and B. Courcelle [31, 8]. By means of a slight modification of the notion of branch, we obtain a very fruitful tool for our purpose. But we need a lot of technical definitions and results.

Definition 1 - Let t be in $M^{\infty}_{A}(V)$, the set br(t) of branches of t is the subset of $W^{*}_{A}(A \cup V)$ equal to $\{w.t(w) \mid w \in dom(t)\}$.

By definition of the syntactic order over $M^{\infty}_{A}(V)$ we have, for any pair (t,t') of elements of $M^{\infty}_{A}(V)$, $t \sqsubseteq t'$ if and only if $br(t) \subseteq br(t')$.

Lemma 1 - Let t, t_1, ..., t_p be elements of $M^{\infty}_{A}(V)$, $\{x_{i_j} \mid j \in [p]\}$ be the set of variables occurring in t, then

$$br(t[t_1/x_{i_1},\dots,t_p/x_{i_p}]) = (br(t) - \bigsqcup_{j\in[p]} W_{i_j}.x_{i_j}) \cup \bigsqcup_{j\in[p]} W_{i_j}.br(t_j),$$

where $W_{i_j} = \{w \in dom(t) \mid t(w) = x_{i_j}\} = \{w \mid w.x_{i_j} \in br(t)\}$.

Now we could construct the context free grammar G_Σ associated to a RPS Σ and generating the language of branches. If Σ is described by

$$\begin{cases} \varphi_i x_1 \dots x_{n_i} = \tau_i \\ 1 \leqslant i \leqslant N \end{cases}$$

then G_Σ is given by the triple ($\mathfrak{T}$, X, P) where

$\mathfrak{T}$ is the nonterminal symbols' alphabet and equal to $\Phi \cup W_\Phi$,

X is the terminal symbols' alphabet and equal to $W_A \cup A \cup V$,

P is the set of productions, included in $\mathfrak{T} \times (\mathfrak{T} \cup X)^*$, and defined by

$$(\varphi_i \to w.f) \in P \text{ iff } w.f \in br(\tau_i) \cap W^*_{A\cup\Phi}.(A\cup\Phi)$$
$$((\varphi_i,j) \to w) \in P \text{ iff } w.x_j \in br(\tau_i) \cap W^*_{A\cup\Phi}.V$$

Example - (example of §1 continued)

If Σ is the RPS described in the example of §1, the grammar G is given by :

$\mathfrak{T} = \{\varphi_1, \varphi_2, \varphi_3\} \cup \{(\varphi_1,1), (\varphi_2,1), (\varphi_3,1)\}$

$X = \{(a,1),(a,2)\} \cup \{a\} \cup \{x\}$

P is the following set of productions

$$\varphi_1 \to a + (a,2)\varphi_2 + (a,2)(\varphi_2,1)\varphi_1$$

$$\varphi_2 \to a + (a,1)\varphi_1 + (a,2)\varphi_2 + (a,2)(\varphi_2,1)\varphi_2$$

$$\varphi_3 \to a + (a,2)\varphi_3 + (a,2)(\varphi_3,1)\varphi_3 + (a,2)(\varphi_3,1)(\varphi_3,1)\varphi_3$$

$$(\varphi_1,1) \to (a,1) + (a,2)(\varphi_2,1)(\varphi_1,1)$$

$$(\varphi_2,1) \to (a,1)(\varphi_1,1) + (a,2)(\varphi_2,1)(\varphi_2,1)$$

$$(\varphi_3,1) \to (a,1) + (a,2)(\varphi_3,1)(\varphi_3,1)(\varphi_3,1)$$

Definition 2 [17] - Let G, (X, Ξ, P) be a context-free grammar, G is said strict deterministic if there exists a partition over $X \cup \Xi$ such that

(i) X is exactly a class of η,

(ii) $\forall \xi, \xi' \in \Xi$, $\forall \alpha, \beta, \beta' \in (X \cup \Xi)$ such that $\xi \to \alpha\beta$ and $\xi' \to \alpha\beta'$ belong to P, if ξ and ξ' are in the same class of η, then the first letter of β and the first letter of β' are in the same class or β and β' are empty and $\xi = \xi'$.

Proposition 1 - Let Σ be a RPS and G_Σ be the context-free grammar associated then G_Σ is a strict deterministic grammar.

Proof - We only give the partition η over $X \cup \Xi$ defined by

$$X \cup \{\{\varphi_i\} \cup \{(\varphi_i, j)\} \mid 1 \leqslant j \leqslant n_i\} \mid 1 \leqslant i \leqslant N\}$$

Definition 3 - Let Σ, (A, Φ, R) be a RPS and t be an element of $\overline{M}_{A\cup\Phi}(V)$, the language $L_\Sigma(t)$ of branches generated by t in G_Σ (as defined above) is the set $\{w \mid w \in X^*$ and $\exists u \in br(t) : u \xrightarrow[P]{*} w\}$, included in $W_A^*.(A \cup V)$ actually.

We need more technical definitions. Let Σ, (A, Φ, R), be a RPS and G_Σ be the context-free grammar associated to Σ, we introduce the following applications

- the language substitution β_Σ such that $\beta_\Sigma(x) = \{x\}$, for any x of X, and $\beta_\Sigma(\xi) = \{w \mid \xi \xrightarrow[P]{} w\}$ for any ξ of Ξ;

- γ_Σ a mapping from $(X \cup \Xi)^*$ to the set of languages over $(X \cup \Xi)^*$ such that, for any w in $(X \cup \Xi)^*$, either w belongs to X^* and $\gamma_\Sigma(w) = \{w\}$ or there exists u in X^*, ξ in Ξ, w' in $(X \cup \Xi)^*$ such that $w = u\,\xi\,w'$ and $\gamma_\Sigma(w) = \{uvw' \mid \xi \xrightarrow[P]{} v\}$;

- ι an application such that, for any L subset of $(X \cup \Xi)^*$, $\iota(L)$ is the subset equal to $L \cap X^*$.

From these definitions we reach the following lemma giving relations between β_Σ, γ_Σ, $\sim$ and α_Σ, σ_Σ and π by means of br.

Lemma 2 - Let Σ, (A, Φ, R), be a RPS, we have the following identities

(i) $\beta_\Sigma \circ br = br \circ \sigma_\Sigma$

(ii) $\gamma_\Sigma \circ br = br \circ \alpha_\Sigma$

(iii) $\imath \circ br = br \circ \pi$

Proof (i) is proved by structural induction on t, element of $\overline{M}_{A\cup\Phi}(V)$. We shall show that, for any t of $\overline{M}_{A\cup\Phi}(V)$, $\beta_\Sigma \circ br(t) = br \circ \sigma_\Sigma(t)$; if t belongs to V then $\sigma_\Sigma(t) = t$, $br(t) = \{t\}$ and $\beta_\Sigma(t) = \{t\}$; if t does not belong V, either $t(\Lambda)$ is in A (a) or $t(\Lambda)$ is in Φ (b).

(a) $t(\Lambda) = a$, $a \in A_k$; for every j in [k] let us note t_j the term $t/(a,j)$: br(t) is equal to $\{a\} \cup \bigsqcup_{j\in[k]} (a,j).br(t_j)$ thus $\beta_\Sigma \circ br(t)$ is equal to $\{a\} \cup \bigsqcup_{j\in[k]} (a,j).\beta_\Sigma \circ br(t_j)$;

by induction we have

$$\beta_\Sigma \circ br(t) = \{a\} \cup \bigsqcup_{j\in[k]} (a,j).br \circ \sigma_\Sigma(t_j) = br(a\sigma_\Sigma(t_1)\ldots\sigma_\Sigma(t_k)),\text{ then}$$

$\beta_\Sigma \circ br(t)$ is equal to $br \circ \sigma_\Sigma(t)$.

(b) $t(\Lambda) = \varphi_i$, $\varphi_i \in \Phi$; for every j in $[n_i]$ let us note t_j the subterm $t/(\varphi_i,j)$: br(t) is equal to $\{\varphi_i\} \cup \bigsqcup_{j\in[n_i]} (\varphi_i,j).br(t_j)$

then $\beta_\Sigma \circ br(t) = \beta_\Sigma(\varphi_i) \cup \bigsqcup_{j\in[n_i]} \beta_\Sigma((\varphi_i,j).br(t_j))$

but $\beta_\Sigma(\varphi_i) = br(\tau_i) \cap W^*_{A\cup\Phi}.(A\cup\Phi)$

and $\beta_\Sigma((\varphi_i,j).br(t_j)) = \{w \mid w.x_j \in br(\tau_i)\}.\beta_\Sigma \circ br(t_j) =$

$= \{w \mid w.x_j \in br(\tau_i)\}.br \circ \beta_\Sigma(t_j)$

Since $\sigma_\Sigma(t)$ is equal to $\tau_i[\sigma_\Sigma(t_1)/x_1,\ldots,\sigma_\Sigma(t_{n_i})/x_{n_i}]$ by lemma 1, we obtain $\beta_\Sigma \circ br(t) = br \circ \sigma_\Sigma(t)$.

(ii) is proved in a very similar way.

(iii) is also proved by structural induction on $\overline{M}_{A\cup\Phi}(V)$, let t be in $\overline{M}_{A\cup\Phi}(V)$:

if t belongs to V then $\pi(t) = t$, $br(t) = \{t\} \subseteq X^*$; if not $t(\Lambda)$ is either in A (a) or in Φ (b).

(a) $t(\Lambda) = a$, $a \in A_k$; for every j in [k] let us note t_j the term $t/(a,j)$ then $\imath \circ br(t) = \{a\} \cup \bigsqcup_{j\in[k]} (a,j).\imath \circ br(t_j)$ and, by induction

$br(t) = \{a\} \cup \bigsqcup_{j\in[k]} (a,j).br \circ \imath\, (t_j) = br \circ \pi(t)$ since $\pi(t)$ is equal to $a\pi(t_1)\ldots\pi(t_k)$.

(b) $t(\Lambda) = \varphi_i$, $\varphi_i \in \Phi$; then $\pi(t)$ is equal to Ω and $br \circ \pi(t)$ is empty. For each j in $[n_i]$, let us not t_j the term $t/(\varphi_i,j)$ then $br(t)$ equals to $\{\varphi_i\} \cup \bigsqcup_{j\in[n_i]} (\varphi_i,j).br(t_j)$ and $\imath \circ br(t) = br(t) \cap X^* = \emptyset$.

As a corollary, we get the following relations

$\imath \circ \beta_\Sigma^n \circ br(t) = br \circ \pi \circ \sigma_\Sigma^n(t)$ and $\imath \circ \gamma_\Sigma^n \circ br(t) = br \circ \pi \circ \alpha_\Sigma^n(t)$ for any t in $\overline{M}_{A\cup\Phi}(V)$ and any integer n.

<u>Proposition 2</u> - Let Σ, (A, Φ, R), be a RPS and t be an element of $\overline{M}_{A\cup\Phi}(V)$, then $L_\Sigma(t)$ is equal to $br(t_\Sigma)$.

<u>Proof</u> - We know by proposition 2 of §2 that t_Σ is the least upperbound of the set $\{\pi \circ \sigma_\Sigma^n(t) \mid n \in N\}$; that means $br(t_\Sigma) = \bigsqcup_{n\in N} br \circ \pi \circ \sigma_\Sigma^n(t)$. By the above relations we reach the equality $br(t_\Sigma) = \bigsqcup_{n\in N} \imath \circ \beta_\Sigma^n \circ br(t)$. But we know, by a theorem of Schützenberger [33], that $L_\Sigma(t)$ is equal to $\bigsqcup_{n\in N} \imath \circ \beta_\Sigma^n \circ br(t)$.

As a corollary we get

<u>Theorem 3</u> (see also [8])

Let Σ, (A, Φ, R), be a RPS and (s,s') be a pair of elements of $\overline{M}_{A\cup\Phi}(V)$; then $s \leq_\Sigma s'$ holds if and only if $L_\Sigma(s)$ is included in $L_\Sigma(s')$.

<u>Proof</u> - We know (by proposition 1 of §3) that $s \leq_\Sigma s'$ if and only if $s_\Sigma \sqsubseteq s'_\Sigma$ holds. By definition of the syntactic order and branches, we know $s_\Sigma \sqsubseteq s'_\Sigma$ holds if and only if $br(s_\Sigma)$ is included in $br(s'_\Sigma)$.

IV. Recursion induction principle. Classes of valid formulas.

§1 Recursion induction principle

In the sequel ρ denotes a (universally) correct computation rule (cf. III. §2 in fine), $\mathfrak{F}$ denotes a set of functions from N into itself, Σ, (A, Φ, R), is a RPS and S is a subset of $M_A(V) x M_A(V)$; now we state our induction principle related to $(\rho, \mathfrak{F})$:

Let R be a subset of $M_{A\cup\Phi}(V) x M_{A\cup\Phi}(V)$ equal to $\{(t_j, t'_j) \mid j \in J\}$; if for each j in J there exists f_j in $\mathfrak{F}$ such that

$$\forall n \in N \quad \pi \circ \rho_\Sigma^n(t_j) \xrightarrow[R_n]{*} \pi \circ \rho_\Sigma^{f_j(n)}(t_j)$$

where R_n is the subset $S \cup \{(\pi \circ \rho_\Sigma^i(t_j), \pi \circ \rho_\Sigma^{f_j(i)}(t'_j)) \mid j \in J, i < n\}$ of $M_A(V)^2$; then the inequality $t_j \leqslant_{<\Sigma,S>} t'_j$ holds for each j in J or, in other words, R is included in $\leqslant_{<\Sigma,S>}$.

If Σ and S are given we shall say that R is provable by a $(\rho, \mathfrak{F})$-induction. To prove the validity of the induction principle we need the following fact

<u>Fact</u> - Let (t,t') be a pair of elements of $M_A(V)$ such that $t \xrightarrow[S]{*} t'$; if S' is equal to $S \cup \{(t,t')\}$ then we have the identity $\sqsubseteq_S = \sqsubseteq_{S'}$.

<u>Proposition 1</u> - The recursion induction principle is valid.

<u>Proof</u> - The above proposition implies that for any integer n, $\sqsubseteq_{R_0}$ is equal to $\sqsubseteq_{R_n}$. But R_0 is equal to S and the proposition $\forall n \in N \; \pi \circ \rho_\Sigma^n(t_j) \xrightarrow[R]{} \pi \circ \rho_\Sigma^{f_j(n)}(t'_j)$ may be rephrased in $\forall n \in N \; \pi \circ \rho_\Sigma^n(t_j) \sqsubseteq_S \pi \circ \rho_\Sigma^{f_j(n)}(t'_j)$ and the theorem of Guessarian (theorem 2) ensures that the inequality $t_j \leqslant_{<\Sigma,S>} t'_j$ holds for each j in J.

The interest of this new statement of recursion induction lies not only in the definition of an <u>effective</u> induction proof but also in the definition of a complexity measure of proofs and a criterion to decide whether a proof is <u>better</u> than another one. This suggests to define "complexity" classes of atomic formulas.

We note $I_\Sigma^S(\rho, \mathfrak{F})$ the set of valid atomic formulas which is a subset of $M_{A\cup\Phi}(V) x M_{A\cup\Phi}(V)$ defined by

$$(t,t') \in I^S_\Sigma(\rho,\mathfrak{F}) \text{ iff } \exists f \in \mathfrak{F}\ \forall n \in N\ \pi \circ \rho^n_\Sigma(t) \sqsubseteq_S \pi \circ \rho^{f(n)}_\Sigma(t')$$

Obviously, each formula of $I^S_\Sigma(\rho,\mathfrak{F})$ is valid with respect to $\leqslant_{\langle\Sigma,S\rangle}$

Problem - Let ρ be a correct computation rule (which will always be σ or α) and S be a subset of $M_A(V)xM_A(V)$; for what class of functions from N into N, say $\mathfrak{F}$, do we have the inclusion $\leqslant_{<\Sigma,S>} \subseteq I^S_\Sigma(\rho,\mathfrak{F})$ for any RPS Σ.

This question is very important since its answer allows to study the completeness of a formal proof system by finding the set such that $I^S_\Sigma(\rho,\mathfrak{F})$ contains formulas provable by the system for any RPS Σ.

To give some answers we distinguish the case where S is the empty set (§2) and where it is not (§3).

§2 The complexity of syntactic inequality

In this paragraph we are going to show that, for any RPS Σ, we have the identities $I_\Sigma(\alpha,\text{Lin}) = \leqslant_\Sigma$ and $I_\Sigma(\sigma,\text{Exp}) = \leqslant_\Sigma$. In other words we are able to bound the complexity of strong inequality by the linear (resp. exponential) functions from N to N.

We call Lin the subset of functions from N to N equal to $\{f \mid \exists a, b \in N\ \forall n \in N\ f(n) = a.n + b\}$ and Exp the subset of functions from N to N equal to $\{f \mid \exists a, b, c \in N\ \forall n \in N\ f(n) = a.b^n + c\}$. We shall say that Lin is "included" in Exp because any function of Lin may be bounded by an element of Exp.

In the section 4 of part III we introduce G_Σ, (X,Ξ, P), the context-free grammar of branches associated to a RPS Σ ; to show the result claimed above we need some technical remarks.

Definition 1 - Let G_Σ, (X,Ξ, P), the grammar associated to a given RPS Σ ; let us define the sequence

$$\Xi_0 = \{\xi \in \Xi \mid \exists w \in X^*\ \xi \longrightarrow w \in P\}$$

$$\Xi_{n+1} = \{\xi \in \Xi \mid \exists w \in (X \cup \Xi_n)^*\ \xi \longrightarrow w \in P\}$$

then $\overline{G}_\Sigma$, $(X, \overline{\Xi}, \overline{P})$ defined by $\overline{\Xi} = \bigsqcup_{n\in N}\Xi_n$ and $\overline{P} = P \cap \overline{\Xi}x(X\cup\overline{\Xi})^*$ is the standard "reduced" grammar associated to G_Σ.

Proposition 1 - [17] Let $\overline{G}_\Sigma$ be the reduced grammar defined above, then the two following propositions hold

$$\forall \xi \in \Xi \quad \xi \in \overline{\Xi} \iff \{w \in X^* \mid \xi \xrightarrow[P]{*} w\} \neq \emptyset$$

and for any finite subset Y of $(X \cup \Xi)$

$$\{w \in X^* \mid \exists u \in Y \; u \xrightarrow[P]{*} w\} = \{w \in X^* \mid \exists u \in Y \cap (X \cup \overline{\Xi})^* \; u \xrightarrow[P]{*} w\}$$

Proposition 2 - Let $\overline{G}_\Sigma$ be the reduced grammar defined above, then $\overline{\Xi}$ does not contain any left recursive symbol i.e. symbol ξ such that $\xi \xrightarrow[P]{+} \xi$ w for some word w.

This is because G_Σ is a strict deterministic grammar in the sense of M. Harrison and I. Havel (cf. §4 in part III).

Lemma 1 - Let $\overline{G}_\Sigma$ be the reduced grammar, then there exists an integer P such that, for any w in $(X \cup \overline{\Xi})^*$ and u in X^* with $w \xrightarrow[P]{*} u$, there exists a leftmost derivation from w to u of length less than $p.|u|$ ($|u|$ is the length of the word u).

Proof - For any ξ in $\overline{\Xi}$ and w in $\overline{\Xi}\,(X \cup \overline{\Xi})^*$, if there is a leftmost derivation from ξ to w its length is less than p' where p' denotes the number of elements of $\overline{\Xi}$ plus one by the proposition 2 above. Let p be p'+1 and we reach the result.

Theorem 4

For any RPS Σ, (A, Φ, R), we have the following identities (i) $\leqslant_\Sigma = I_\Sigma(\alpha,\mathrm{Lin})$ and (ii) $\leqslant_\Sigma = I_\Sigma(\sigma,\mathrm{Exp})$.

Proof - It suffices to show the two inclusions

(i) $\leqslant_\Sigma \subseteq I_\Sigma(\alpha,\mathrm{Lin})$ and (ii) $\leqslant_\Sigma \subseteq I_\Sigma(\sigma,\mathrm{Exp})$ (completeness property). For any integer n and t in $M_{A\cup\Phi}(V)$, we shall note $\theta_n^\Sigma(t)$ the tree of $M_A(V)$ defined by the set of its branches $br(\theta_n^\Sigma(t)) = \{u \mid u \in br(t_\Sigma) \text{ and } |u| \leqslant n\}$. First, we prove the inclusion (i).

Fact 1 - Let us consider some u belonging to $br(t_\Sigma)$, then the proposition 2 of III.§4 ensures us that u belongs to $L_\Sigma(t)$; thus there exists an element w in br(t) such that $w \xrightarrow[P]{*} u$ and, by the lemma 1, we know there exists a leftmost derivation from w to u of length less than $p.|u|$ (p is the integer of lemma 1).

This fact and the definition of the application γ_Σ (cf. definition in III.§4) involve that u belongs to $\gamma_\Sigma^{p.|u|} \varepsilon$ br(t). Thus we have the relation $br(\theta_n^\Sigma(t)) = \imath \circ br(\theta_n(t)) \subseteq \imath \circ \gamma_\Sigma^{p.n} \circ br(t)$; applications of results of III.§4 give us $br(\theta_n(t)) \subseteq \imath \circ br \circ \alpha_\Sigma^{p.n}(t) \subseteq br \circ \pi \circ \alpha_\Sigma^{p.n}(t)$ which is equivalent to $\theta_n^\Sigma(t) \sqsubseteq \pi \circ \alpha_\Sigma^{p.n}(t)$.

Fact 2 - For any t in $M_{AU\Phi}(V)$ any integer n and any u in $\imath \circ \gamma_\Sigma^n \circ br(t)$ we have the relation $|u| \leq k.n + ||t||$ where $k = \max \{|w| \mid \exists \xi \in \overline{\Xi}\ \xi \xrightarrow[P]{} w\}$ and $||t|| = \max \{|u| \mid u \in br(t)\}$ (by convention $||\Omega|| = 0$).
But $br \circ \pi \circ \alpha_\Sigma^n(t)$ is equal to $\imath \circ \gamma_\Sigma^n \circ br(t)$ and any branch of $\pi \circ \alpha_\Sigma^n(t)$ is a branch of t_Σ such that its length is less than $k.n + ||t||$, so it is a branch of $\theta^\Sigma_{k.n+||t||}(t)$. We reach the inclusion $br \circ \pi \circ \alpha_\Sigma^n(t) \subseteq br(\theta^\Sigma_{k.n+||t||}(t))$ and the inequality $\pi \circ \alpha_\Sigma^n(t) \sqsubseteq \theta^\Sigma_{k.n+||t||}(t)$.

From Fact 1 and Fact 2 we deduce that for any pair (s,s') of elements of $M_{AU\Phi}(V)$ such that $s \leq_\Sigma s'$ we have

$$\pi \circ \alpha_\Sigma^n(s) \sqsubseteq \theta^\Sigma_{k.n+||s||}(s) \sqsubseteq \theta^\Sigma_{k.n+||s||}(s') \sqsubseteq \pi \circ \alpha_\Sigma^{p.k.n+p.||s||}(s')$$

thus (s,s') is in $I_\Sigma(\alpha,\mathrm{Lin})$. Now we prove the inclusion (ii).

Fact 3 - By an obvious induction on integers we have the logical implication : $u \in \imath \circ \beta_\Sigma^n \circ br(t) \Rightarrow |u| \leq k^n.||t||$ for any t of $M_{AU\Phi}(V)$. By a result of III.§4 and Fact 1 we reach the inequality

$$\pi \circ \sigma_\Sigma^n(t) \sqsubseteq \pi \circ \alpha_\Sigma^{p.||t||.k^n}(t).$$

Fact 4 - Since $\alpha_\Sigma^n(t) \xrightarrow[\Sigma]{*} \sigma_\Sigma^n(t)$ we have also $\pi \circ \alpha_\Sigma^n(t) \sqsubseteq \pi \circ \sigma_\Sigma^n(t)$. Combining Fact 3 and Fact 4, we deduce that $s \leq_\Sigma s'$ implies, for any integer n,

$$\pi \circ \sigma_\Sigma^n(s) \sqsubseteq \pi \circ \alpha_\Sigma^{p'.k^n}(t) \sqsubseteq \pi \circ \alpha_\Sigma^{\overline{p}.k^n+\overline{q}}(s') \sqsubseteq \pi \circ \sigma_\Sigma^{\overline{p}.k^n+\overline{q}}(s')$$

with $p' = p.||s||$, $\overline{p} = p^2.k.||s||$, $\overline{q} = q^2.||s||$; thus (s,s') belongs to $I_\Sigma(\sigma,\mathrm{Exp})$.

Proposition 3 - (a particular case of incompleteness). The identity

$$\leq_\Sigma = I_\Sigma(\sigma,\mathrm{Lin}) \text{ does not hold for any RPS } \Sigma.$$

It suffices to exhibit a counter-example of two elements, t and t', such that $t \leq_\Sigma t'$ and (t,t') does not belong to $I_\Sigma(\sigma,\mathrm{Lin})$. Let us recall the RPS described in the example of part III

$$\Sigma \begin{cases} \varphi_1 x = ax\varphi_2\varphi_1 x \\ \varphi_2 x = a\varphi_1 x\varphi_2\varphi_2 x \\ \varphi_3 x = ax\varphi_3\varphi_3\varphi_3 x \end{cases}$$

we have given already the associated grammar

$$\varphi_1 \to a + (a,2)\varphi_2 + (a,2)(\varphi_2,1)\varphi_1$$

$$\varphi_2 \to a + (a,1)\varphi_1 + (a,2)\varphi_2 + (a,2)(\varphi_2,1)\varphi_2$$

$$\varphi_3 \to a + (a,2)\varphi_3 + (a,2)(\varphi_3,1)\varphi_3 + (a,2)(\varphi_3,1)\varphi_3$$

$$(\varphi_1,1) \to (a,1) + (a,2)(\varphi_2,1)(\varphi_1,1)$$

$$(\varphi_2,1) \to (a,1)(\varphi_1,1) + (a,2)(\varphi_2,1)(\varphi_2,1)$$

$$(\varphi_3,1) \to (a,1) + (a,2)(\varphi_3,1)(\varphi_3,1)(\varphi_3,1)$$

On $\{(a,1),(a,2)\}^*$ we define two sequences $(u_n)_{n\geq 1}$ and $(v_n)_{n\geq 1}$ by induction :

$$u_1 = (a,1)$$

$$u_{n+1} = (a,2)u_n u_n u_n$$

$$v_n = u_n u_n$$

We know G_Σ is strict deterministic, thus for any non terminal symbol ξ the set $\{u \mid u \in X^* \ \xi \xrightarrow[P]{*} u\}$ is prefix-free (see [17]). From this remark we are able to prove by induction that, for any integer greater than 1 :

$$u_n \in \imath \circ \beta_\Sigma^n((\varphi_3,1)),\ v_n \in \imath \circ \beta_\Sigma^n((\varphi_3,1)(\varphi_3,1))$$

there exists p such that $u_n \in \imath \circ \beta_\Sigma^p((\varphi_1,1)$ and $p \geq \frac{n^2+n}{2}$

there exists p such that $v_n \in \imath \circ \beta_\Sigma^p((\varphi_2,1)$ and $p \geq \frac{n^2+3n}{2}$

(for this last relation let us remark that $u_n(\varphi_1,1) \in \beta_\Sigma^p((\varphi_2,1))$ with $p \gg n$).

This example is borrowed from [12] who proved the equivalences $\varphi_1 x =_\Sigma \varphi_3 x$ and $\varphi_2 x =_\Sigma \varphi_3\varphi_3 x$, but $(\varphi_3 x, \varphi_1 x)$ does not belong $I_\Sigma(\sigma,\mathrm{Lin})$.

We shall say a RPS Σ is linear (or non nested) if and only if the associated context-free grammar G_Σ is linear.

Proposition 4 - Let Σ be a linear RPS, t and t' two linear trees of $M_{A\cup\Phi}(V)$ (i.e. trees such that each u in br(t) or br(t') contains at most one occurrence of an element W_Φ), then we have

$$t \leqslant_\Sigma t' \Rightarrow (t,t') \in I_\Sigma(\sigma,\mathrm{Lin})$$

Proof - It is obvious since, in this case, $\pi \circ \alpha_\Sigma^n(t)$ (resp. $\pi \circ \alpha_\Sigma^n(t')$) is equal to $\pi \circ \sigma_\Sigma^n(t)$ (resp. $\pi \circ \sigma_\Sigma^n(t')$).

In some sense we give a new reason to call linear this kind of RPSs.

An interesting consequence of Theorem 4 is to show the advantage of the α-induction to prove syntactic inequalities (or equivalences) since proofs are less difficult. So we enforce the point of view developped by G. Boudol [4].

To finish this section we must mention we actually do not know what is the "best" (i.e. minimal with respect to inclusion) class of functions $\mathfrak{F}$ such that $\leqslant_\Sigma$ is equal to $I_\Sigma(\sigma,\mathfrak{F})$ for any RPS Σ. The above example shows that $\mathfrak{F}$ is included in Exp and contains Pol (the set of polynomial functions). Is there any example of strong inequality whom the proof requires exponential σ-induction ? We do not believe and we state the following

Conjecture 1 - Let Pol be the set of functions from N to N $\{f \mid \exists a_0,\dots,a_p \in N$ $f(n) = \sum_{i=0}^{p} a_i n^i\}$, then we have the identity $\leqslant_\Sigma = I_\Sigma(\sigma,\mathrm{Pol})$ for any RPS Σ.

§3 General case

Let us recal we would find the class $\mathfrak{F}$ of function from N to N such that the inclusion $\leqslant_{<\Sigma,S>} \subseteq I_\Sigma^S(\rho,\mathfrak{F})$ for any RPS Σ; a computation rule ρ and a non empty set S given. In this case the situation is more complicated and we would like brought to light three remarks :

(1) In the introduction we give the following example (due to R. Milner) let Σ be the RPS

$$\begin{cases} \varphi x = f\varphi x \\ \psi x = g\psi x \end{cases}$$

and S be the subset of $M_{\{f,g\}}(\{x\})^2$ $\{(fgx,g^2fx),(f\Omega,f\Omega)\}$ then we have the equivalence $\varphi x \equiv_{<\Sigma,S>} \psi x$ and, for any integer n, the relation $\pi \circ \sigma_\Sigma^n(\varphi x) \sqsubseteq_S \pi \circ \sigma_\Sigma^p(\psi x)$ implies p greater than 2^n-1.

From this example, we deduce that $\leqslant_{<\Sigma,S>}$ is included neither in $I_\Sigma^S(\sigma,\mathrm{Pol})$ nor in $I_\Sigma^S(\alpha,\mathrm{Pol})$. So the desired class $\mathcal{F}$ is at least Exp.

(2) If we call All the class of all functions from N to N, we have the obvious property : for any RPS Σ, for any finite subset S of $M_A(V)^2$, $\leqslant_{<\Sigma,S>}$ is equal to $I_\Sigma^S(\rho,\mathrm{All})$ (ρ is either α or σ).

(3) It is well-known it is possible to construct a finite system of terms rewriting, say S, such that it encodes the behaviour of a Turing machine ; so we know that the class $\mathcal{F}$ must contain the recursive functions and we state the following

Conjecture 2 – The least class of functions from N to N, say $\mathcal{F}$, such that, for any RPS Σ, any finite subset S of $M_A(V)^2$, $\leqslant_{<\Sigma,S>}$ is equal to $I_\Sigma^S(\rho,\mathcal{F})$ (ρ is either α or σ) is the class of recursive functions.

Thus a good question may be for some class of particular functions, say $\mathcal{F}$, could we give some properties such that if S satisfies them the inclusion $\leqslant_{<\Sigma,S>} \subseteq I_\Sigma^S(\rho,\mathcal{F})$ holds. By this way we could reach a tool to define a hierarchy over the finite systems of terms rewriting.

But our recursion induction principle may be also used as mean to study the validity and the completeness of formal systems to prove inequalitites or equivalences. For example in [23] we proved that the so-called fold/unfold method, elaborated by R. Burstall and J. Darlington [6], is an incomplete method for proving programs equivalences. Let us give some explanations : a system for performing transformations of recursive programs (such as the Burstall-Darlington system) may be viewed as a system for proving programs equivalence since one can perform a transformation from P_1 to P_2 which preserves the equivalence (under some assumptions, see [9, 23, 24]) one prove the equivalence between P_1 and P_2.

The next paragraph is devoted to the presentation of a powerful formal system for proving recursive program schemes inequalities and equivalences.

§4 A formal system for proving inequalities in RPS's

We take our inspiration in [12] to design a formal system, about which we prove our main result : for all proof in the system we define, in a constructive way, a recursive function which is a bound for the complexity of the proved formula. This is shown when the axioms are equations over the A-magma $M_A(V)$, that is are elements of a symetric binary relation which expresses properties of base functions, and which we always denote S.

Here we assume the existence of a procedure symbols universe Ψ, so that each RPS $\Sigma = (A, \Phi, R)$ satisfies $\Phi \subseteq \Psi$ and for all $\varphi \in \Phi$, of arity n, there exists exactly one $\tau \in M_{A \cup \Psi}(\{x_1, \ldots, x_n\})$ such that $(\varphi x_1 \ldots x_n, \tau) \in R$.

Thus, the elements of Φ are exactly the procedure symbols defined by Σ, and Σ left other symbols of Ψ undefined, although they can occur in the body of some definition in Σ. We call Alph(Σ) (the alphabet of procedure symbols of Σ) the set of procedure symbols which occur in some definition in Σ (thus $\Phi \subseteq$ Alph(Σ)). Two RPS's $\Sigma = (A, \emptyset, \Sigma)$ and $\Sigma' = (A, \Phi', R')$ are <u>independant</u> iff Σ does not define an element of the alphabet of Σ', and conversely, ie $\Phi \cap \mathrm{Alph}(\Sigma') = \emptyset = \Phi' \cap \mathrm{Alph}(\Sigma)$ and in this case $\Sigma \cup \Sigma' = (A, \Phi \cup \Phi', R \cup R')$ is also an RPS.

One can easily check that in this case σ_Σ and $\sigma_{\Sigma'}$ commute : $\sigma_\Sigma \circ \sigma_{\Sigma'} = \sigma_{\Sigma'} \circ \sigma_\Sigma = \sigma_{\Sigma \cup \Sigma'}$.

We also make more precise the notion of immediate information relative to a RPS $\Sigma = (A, \Phi, R)$, which is the projection π_Σ on $M_{A \cup \Psi}(V)$ defined as π, except for :

$$\pi_\Sigma(\psi m_1 \ldots m_k) = \begin{cases} \Omega \text{ if } \psi \in \Phi \\ \psi \pi_\Sigma(m_1) \ldots \pi_\Sigma(m_k) \text{ otherwise } (\psi \in \Psi) \end{cases}$$

Thus if Σ and Σ' are independant :

$$\pi_\Sigma \circ \pi_{\Sigma'} = \pi_{\Sigma'} \circ \pi_\Sigma = \pi_{\Sigma \cup \Sigma'}$$

The formal system is designed to prove <u>formulas</u> : an atomic formula is an inequality $t \subseteq t'$ where t and t' belong to $M_{A \cup \Psi}(V)$ (ie are terms with Ω as a symbol of constant) and a formula is a finite conjunction of atomic formulas. We often identify such a formula with a finite binary relation over $M_{AU\Psi}(V)$, writen $\{t_1 \subseteq t'_1, \ldots, t_p \subseteq t'_p\}$, and abreviate the conjunction of $t \subseteq t'$ and $t' \subseteq t$ by $t = t'$ (a symetric relation).

The (procedure symbols) alphabet of such a formula P is the (finite) set Alph (P) of element of Ψ occurring in some atomic inequality of P, and we say that P and the RPS $\Sigma = (A, \Phi, R)$ are <u>independant</u> iff $Alph(P) \cap \Phi = \emptyset$

In this case $\pi_\Sigma \times \pi_\Sigma(P) = \sigma_\Sigma \times \sigma_\Sigma(P) = P$

We need another notation before describing the system : for a formula P and a RPS $\Sigma = (A, \Phi, R)$, we define the RPS $\Sigma|P$, the restriction of Σ to P as $\Sigma|P = (A, \Phi|P, R|P)$

where $\Phi|P$ is the union of the $\Phi^{(n)}$'s given by

$$\begin{cases} \Phi^{(0)} = \Phi \cap Alph(P) \\ \psi \in \Phi^{(n+1)} \iff \exists \varphi \in \Phi \cap \Phi^{(n)}, \psi \text{ occurs in } \tau \text{ s.t. } (\varphi x_1 \ldots x_n, \tau) \in R \text{ or} \\ \qquad\qquad \exists \varphi \in \Phi^{(n)} : \varphi \text{ occurs in } \tau \text{ s.t. } (\psi x_1 \ldots x_p, \tau) \in R \end{cases}$$

and $R|P = R \cap (M_{\Phi|P}(V) \times M_{AU\Psi}(V))$.

Then $\Sigma - P = (A, \Phi - \Phi|P, R - R|P)$ is also an RPS, independant from $\Sigma|P$ and P, and such that $\Sigma = \Sigma|P \cup (\Sigma - P)$.

The description of the formal system consists in the inductive definition of the notion of the (syntactic) consequence relation relative to an RPS , which is the least relation $\vdash_\Sigma$ between formulas such that :

(1) <u>Replacement rule</u>

if $Q \subseteq (\sqsubseteq \cup \xleftrightarrow[\Sigma]{} \cup \xrightarrow[P]{})^*$ then $P \vdash_\Sigma Q$

(2) <u>Union rule</u> (or conjunction rule)

if $P \vdash_\Sigma Q$ and $P' \vdash_\Sigma Q'$ then $P \cup P' \vdash_\Sigma Q \cup Q'$

(3) <u>Cut rule</u>

if $P \vdash_\Sigma Q'$ and $Q' \vdash_\Sigma Q$ then $P \vdash_\Sigma Q$

(4) <u>Σ''-induction rule</u>

if Σ' and Σ'' are independant, $\Sigma = \Sigma' \cup \Sigma''$ and P and Σ'' are independant,

if $P \vdash_{\Sigma'} \pi_{\Sigma''} \times \pi_{\Sigma''}(Q)$ and $P \cup Q \vdash_{\Sigma'} \sigma_{\Sigma''} \times \sigma_{\Sigma''}(Q)$
then $P \vdash_\Sigma Q$

(5) Restriction rule

if $P \vdash_{\Sigma} Q$ then $P \vdash_{\Sigma \restriction P \cup Q} Q$

As usual, we must define a concept of semantic validity, or more precisely the semantic relation of consequence in Σ $\models_{\Sigma}$ between formulas on A and formulas on $A \cup \Psi$ as :

$$S \models_{\Sigma} P \iff_{def} P \subseteq \leqslant_{<\Sigma,S>}$$

$\iff$ for all interpretation I satisfying S, for all $(t,t') \in P$:

$$t_{<\Sigma,I>} \leqslant_I t'_{<\Sigma,I>}$$

To be of some use, the system must be valid, which means :

$$S \vdash_{\Sigma} P \Rightarrow S \models_{\Sigma} P$$

This is true, as we shall see.

We may add some derived inference rules to simplify the proofs, such as, for example, the inclusion rule (consequence of replacement and cut rules) :

if $P \vdash_{\Sigma} Q'$ and $Q \subseteq Q'$ then $P \vdash_{\Sigma} Q$

We shall abreviate $\emptyset \vdash_{\Sigma} P$ and $P \vdash_{\emptyset} Q$ resp. by $\vdash_{\Sigma'} P$ and $P \vdash Q$.

A formula Q is provable from P (as axiom) in Σ iff $P \vdash_{\Sigma} Q$, and a proof of Q from P in Σ is a finite binary labelled (by statements $Q' \vdash_{\Sigma'} Q''$) tree, such that :

- its root is labelled by $P \vdash_{\Sigma} Q$
- the leaves are labelled by $Q' \vdash_{\Sigma'} Q''$ which are instance of the replacement rule (and we call the proof explicit if each step of $\sqsubseteq \cup \xleftrightarrow[\Sigma']{} \cup \xrightarrow[Q']{}$ to obtain Q" is described)
- for each node which is not a leaf, the label result from its sons by application of an inference rule.

We do not give a formal definition, but illustrate this notion by some examples.

Example 1 - Let Σ be the RPS :

$$\Sigma \begin{cases} \varphi x = f\psi x \\ \psi x = g\psi x \end{cases}$$

and S be the set of axioms :

$$S = \{fgx = gfx,\ f\Omega = g\Omega\}$$

We want to prove $P = \{\varphi x = \psi x\}$ from S in Σ. A proof (tree) is as follows :

$$\begin{tikzcd}
 & & S \vdash_{\Sigma} \varphi x = \psi x & \\
 & S \vdash_{\Sigma} f\psi x = g\psi x \arrow[ur, "\text{cut}"] & & f\psi x = g\psi x \vdash_{\Sigma} \varphi x = \psi x \arrow[ul] \\
S \vdash f\Omega = g\Omega \arrow[ur] & & S \cup \{f\psi x = g\psi x\} \vdash_{\Sigma} fg\psi x = gg\psi x \arrow[ul, "\Sigma\text{-induction}"'] &
\end{tikzcd}$$

Let us explicit the replacement

- obviously $f\Omega \underset{S}{\longleftrightarrow} g\Omega$
- $fg\psi x = \sigma_{\Sigma}(f\psi x) \underset{S}{\longleftrightarrow} gf\psi x \underset{f\psi x = g\psi x}{\longleftrightarrow} gg\psi x = \sigma_{\Sigma}(g\psi x)$
- $\varphi x \underset{\Sigma}{\longleftrightarrow} f\psi x \underset{f\psi x = g\psi x}{\longleftrightarrow} g\psi x \underset{\Sigma}{\longleftrightarrow} \psi x$

This proof is easy, because it only needs an obvious lemma $f\psi x = g\psi x$, which is $\sigma_{\Sigma}(\varphi x) = \sigma_{\Sigma}(\psi x)$ (indeed the formula $\varphi x = \psi x$ belongs to $E^{S}_{\Sigma}(\sigma,\mathrm{Lin})$).

In the following example, the proof needs some more cleaver lemmas, and the introduction of a copy of the given RPS. But let us first precise this notion of copy and its use : a <u>renaming</u> (of procedure symbols) is an application $\rho = \Psi \to \Psi$ which respect the arities ($\rho(\Psi_n) \subseteq \Psi_n$ for all n) and such that there exists a finite subset of Ψ, the support of ρ, denoted $\mathrm{supp}(\rho)$ on which ρ is a transposition (a permutation such that $\rho \circ \rho = \mathrm{id}$), and the identity elsewhere.

We can obviously "extend" such a map (with the same notation) to $M_{A \cup \Psi}(V)$ by

$$\rho(t) = t \text{ if } t \in V \cup \{\Omega\}$$
$$\rho(a^k_n t_1 \dots t_k) = a^k_n \rho(t_1) \dots \rho(t_k)$$
$$\rho(\psi t_1 \dots t_p) = \rho(\psi)\ \rho(t_1) \dots \rho(t_p)$$

and to relations, by $\rho \times \rho$, and to RPS's, if we define for $\Sigma = (A, \Phi, R)$: $\rho(\Sigma) = (A, \rho(\Phi), \rho\times\rho(R))$.
(We can remark that $\rho(\Sigma \cup \rho(\Sigma)) = \Sigma \cup \rho(\Sigma)$).
We call $\rho(\Sigma)$ a <u>copy</u> of Σ iff $\mathrm{supp}(\rho) = \Phi \cup \Phi'$ with $\Phi \cap \Phi' = \emptyset$, $\rho(\Phi) = \Phi'$.

We denote, for a renaming

$E(\rho) = \{\varphi x_1 \ldots x_n = \rho(\varphi)x_1 \ldots x_n / \varphi \in \mathrm{supp}(\rho)\}$

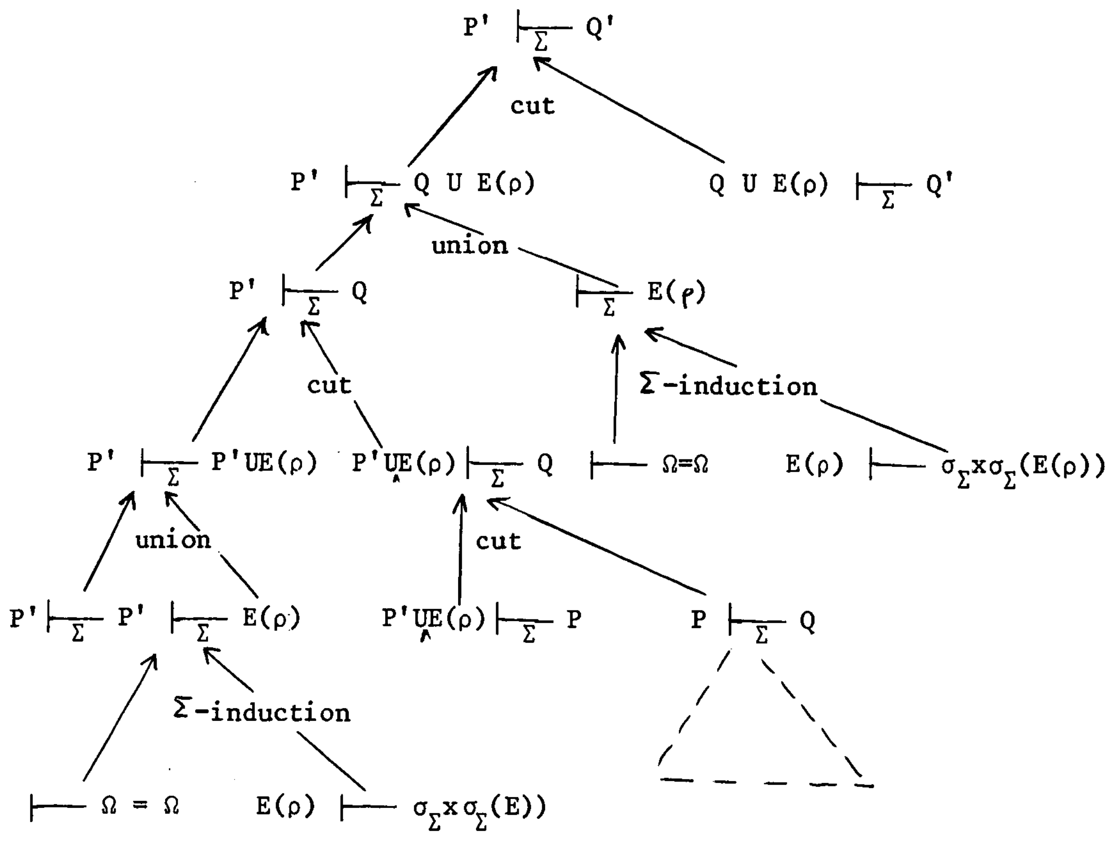
(which states that, up to a renaming, there is no change !) and we say that a formula P' is a renaming, by ρ, of

$P = \{t_i \subseteq t_i' / 1 \leq i \leq p\}$ iff $P' = \{\theta_i \subseteq \theta_i' / 1 \leq i \leq p\}$ and $\forall i\ \theta_i \xleftrightarrow[E(\rho)]{*} t_i$ and $\theta_i' \xleftrightarrow[E(\rho)]{*} t_i'$

(we do not impose that each occurrence of a procedure symbol in the atomic formulas of P is renamed by ρ to get P'). We can then state the

(6) <u>Renaming rule</u>

if $\Sigma = (A, \Phi, R)$ is a RPS, ρ a renaming such that $\mathrm{supp}(\rho) \subseteq \Phi$, if $P \vdash_{\Sigma} Q$ and P' and Q' are resp. renamings of P and Q by ρ, if $\rho(\Sigma) = \Sigma$ then $P' \vdash_{\Sigma} Q'$. In fact, this rule is a derived one, as shown by :



Let us explain the leaves :

. $P' \vdash_{\Sigma} P'$ is a trivial instance of the replacement rule

. $\{\Omega = \Omega\} = \pi_\Sigma \times \pi_\Sigma(E(\rho))$ since $\mathrm{supp}(\rho) \subseteq \Phi$

. $E(\rho) \vdash \sigma_\Sigma \times \sigma_\Sigma(E(\rho))$ since, for $\varphi \in \mathrm{supp}(\rho)$:

$$\sigma_\Sigma(\varphi x_1 \dots x_n) \xleftrightarrow[E(\rho)]{*} \rho(\sigma_\Sigma(\varphi x_1 \dots x_n)) = \sigma_{\rho(\Sigma)}(\rho(\varphi)x_1 \dots x_n) = \sigma_\Sigma(\rho(\varphi)x_1 \dots x_n)$$

(since $\rho(\Sigma) = \Sigma$ and $\forall\theta : \rho(\theta) \xleftrightarrow[E(\rho)]{*} \theta$)

. $P' \cup E(\rho) \vdash_\Sigma P$ since by assumption on P' :

$P \subseteq (\xleftrightarrow[E(\rho)]{*} \circ \xrightarrow[P']{} \circ \xleftrightarrow[E(\rho)]{*})$

. $Q \cup E(\rho) \vdash_\Sigma Q'$ since again $Q' \subseteq (\xleftrightarrow[E(\rho)]{*} \circ \xrightarrow[Q]{} \circ \xleftrightarrow[E(\rho)]{*})$

We mainly use this rule in the case of $\Sigma = \Sigma' \cup \rho(\Sigma')$ where $\rho(\Sigma')$ is a copy of Σ'.

<u>Example 2</u> -

$$\Sigma \begin{cases} \varphi x = f\varphi x \\ \psi x = g\psi x \end{cases} \qquad \Sigma' \begin{cases} \varphi' x = f\varphi' x \\ \psi' x = g\psi' x \end{cases} \qquad \text{(a copy of } \Sigma)$$

(here the renaming ρ is given by $\mathrm{supp}(\rho) = \{\varphi, \psi, \varphi', \psi'\}$ with $\rho(\varphi) = \varphi'$, $\rho(\psi) = \psi'$).

$S = \{fgx = ggfx,\ f\Omega = g\Omega\}$

We want to prove $\varphi x = \psi x$ from S in Σ. An obvious first step is :

$$\begin{tikzcd}[column sep=small] & S \vdash_\Sigma \varphi x = \psi x & \\ & \text{union} & \\ S \vdash_\Sigma \varphi x \subseteq \psi x \arrow[uur] & & S \vdash_\Sigma \psi x \subseteq \varphi x \arrow[uul] \end{tikzcd}$$

Let us detail the proof of $S \vdash_\Sigma \varphi x \subseteq \psi x$ (we left to the reader the (easier) proof of the converse inequality $\psi x \subseteq \varphi x$).

$$
\begin{tikzcd}[column sep=small]
 & S \vdash_{\Sigma} \varphi x \subseteq \psi x & & \\
 & S \vdash_{\Sigma \cup \Sigma'} \varphi x \subseteq \psi x \arrow[u, "\text{restriction}"'] & & \\
 & S \vdash_{\Sigma \cup \Sigma'} \varphi x \subseteq \psi' x \arrow[u, "\text{renaming}"'] & & \\
S \vdash_{\Sigma \cup \Sigma'} f\psi' x \subseteq \psi' x \arrow[ur] & & P = \{f\psi' x \subseteq \psi' x\} \vdash_{\Sigma \cup \Sigma'} \{\varphi x \subseteq \psi' x\} = Q \arrow[ul, "\text{cut}"'] & \\
 & & P \vdash_{\Sigma'} \Omega \subseteq \psi' x \arrow[u] & P \cup Q \vdash_{\Sigma'} f\varphi x \subseteq \psi' x \arrow[ul, "\Sigma\text{-induction}"']
\end{tikzcd}
$$

where

$$\sigma_{\Sigma}(\varphi x) = f\varphi x \xrightarrow[Q]{} f\psi' x \xrightarrow[P]{} \psi' x = \sigma_{\Sigma}(\psi' x)$$

The last step is the proof of the lemma $S \vdash_{\Sigma \cup \Sigma'} f\psi' x \subseteq \psi' x$

If we let $P' = \{f\psi' x \subseteq \psi x\}$:

$$
\begin{tikzcd}[column sep=small]
 & S \vdash_{\Sigma \cup \Sigma'} f\psi' x \subseteq \psi' x & \\
 & S \vdash_{\Sigma \cup \Sigma'} f\psi' x \subseteq \psi x \arrow[u, "\text{renaming}"'] & \\
S \vdash_{\Sigma} f\Omega \subseteq \psi x \arrow[ur] & & S \cup P' \vdash_{\Sigma} fg\psi' x \subseteq \psi x \arrow[ul, "\Sigma'\text{-induction}"']
\end{tikzcd}
$$

where :

- $\pi_{\Sigma'}(f\psi'x) = f\Omega \xleftrightarrow[S]{} g\Omega \sqsubseteq g\psi x \xleftarrow[\Sigma]{} x = \pi_{\Sigma'}(\psi x)$
- $\sigma_{\Sigma'}(f\psi'x) = fg\psi'x \xleftrightarrow[S]{} ggf\psi'x \xrightarrow[P']{} gg\psi x \xleftarrow[\Sigma]{} o \xleftarrow[\Sigma]{} \psi x = \sigma_{\Sigma'}(\psi x)$

Intuitively, this proof is more difficult than the one of example 1. (indeed $\varphi x \subseteq \psi x$ belongs to $I^S_\Sigma(\sigma,\mathrm{Exp})$, as we have already seen).

We must point out that the independance hypothesis in the induction rule are crucial for the validity of the system. For example, with the RPS of example 1 and $S' = \{fgx = gfx\}$ we could prove, relaxing one of these two independance hypothesis, that $S' \vdash_\Sigma \varphi x = \psi x$:

- $S' \vdash_\Sigma \varphi x = \psi x$ by "Σ"-induction" (with $\Sigma' = \Sigma'' = \Sigma$) since
  $\sigma_\Sigma(\varphi x) = f\psi x \xleftrightarrow[\Sigma]{} fg\psi x \xleftrightarrow[S]{} gf\psi x \xleftrightarrow[\Sigma]{} g\varphi x \xleftrightarrow[\varphi x = \psi x]{} g\psi x = \sigma_\Sigma(\psi x)$
- $S' \vdash_\Sigma S' \cup \Sigma$ (trivial instance of the replacement rule) and
  $S' \cup \Sigma \vdash_\Sigma \varphi x = \psi x$ is "proved" exactly as above by "Σ"-induction" with
  $P = S' \cup \Sigma$, $\Sigma'' = \Sigma$ and $\Sigma' = \emptyset$
  But $\varphi x \equiv_{<\Sigma,S'>} \psi x$ is false.

We now state the main theorem, expressing the complexity -with respect to $S \subseteq M_A(V)^2$ and Σ- of formulas P provable from S in Σ. To this aim, let us define the set $\mathcal{O}$ as the least subset of the set of functionnals over N (that is mappings from N^N into itself) such that

- $\underline{suc}$, $\underline{id} \in \mathcal{O}$ where, for $f \in N^N$ and $n \in N$:
  $\underline{suc}(f)(n) = f(n)+1$, $\underline{id}(f)(n) = f(n)$
- if $\psi_1, \psi_2 \in \mathcal{O}$ then
  - $\psi_1 \circ \psi_2 \in \mathcal{O}$
  - $\underline{max}(\psi_1, \psi_2) \in \mathcal{O}$ where $(\underline{max}(\psi_1, \psi_2)(f))(n) = \max(\psi_1(f)(n), \psi_2(f)(n))$
  - $\underline{comp}(\psi_1, \psi_2) \in \mathcal{O}$ where $(\underline{comp}(\psi_1, \psi_2)(f))(n) = \psi_1(f)(\psi_2(f)(n))$
  - $\delta(\psi_1, \psi_2) \in \mathcal{O}$ where $\delta(\psi_1, \psi_2)(f)$ is given by
    $(\delta(\psi_1, \psi_2)(f))(n) = (\lambda(n)(f))(n)$
    $\lambda(0) = \psi_1$
    $\lambda(m+1) = \psi_2 \circ \lambda(m)$

Let $\mathfrak{F}$ be the least subset of N^N which contains the identity function and closed by $\mathcal{O}$ ($f \in \mathfrak{F}$ & $\psi \in \mathcal{O} \Rightarrow \psi(f) \in \mathfrak{F}$). It is easy to check that is a subset of the recursive functions and that for all f of $\mathfrak{F}$:

$\forall n \in N \quad f(n+1) \geqslant f(n) \geqslant n$

<u>Theorem 5</u> - Complexity of provable formulas

For all S, symetric binary relation over $M_A(V)$, for all RPS Σ, for all formula P and for all (explicit) proof of P from S in Σ there (effectively) exists f in $\mathfrak{F}$ such that $P \leq I_\Sigma^S(\sigma,\{f\})$. Furthermore the implication

$S \vdash_{\Sigma} P \Rightarrow P \leq I_\Sigma^S(\sigma, \mathfrak{F})$

holds.

To prove this theorem we need a lot of technical definitions and lemmas, so we postpone a detailed proof to the next paragraph.

An easy consequence of this "complexity theorem" is the following corollary.

<u>Corollary: validity</u>.

The system is valid.

(Let us remark this may be reached in a more standard way, see for example [35]).

Our result on complexity of provable formulas means that, if a formula P is known to be valid in S and Σ with a minimal complexity h, then a proof (if any) of P from S in Σ performed by the system must have a complexity scheme ψ such that ψ (id) is greater or equal than h. Thus if h is not a "small" function, the proof must be difficult.

However this assertion is somewhat vague but we feel that the construction of the claim below exhibits (at least for "non-stupid" proofs) a link between an intuitive notion of "difficulty" of a proof, and the speed of increasing of its complexity. It remains to make these ideas more precise. But to support them, let us mention some facts we know :

- if P is proved without use of induction its complexity is a linear function
- if P is proved with Σ'-induction in which $\Sigma' = \emptyset$, again its complexity is a linear function

- if P is proved with "non-nested" induction (no use of induction to prove the hypothesis of the instances of induction rule) then its complexity is (bounded by) an exponential function.

We may remark that the complexity scheme of a proof is completely determined by the "sketch of the proof" : we call "sketch of the proof" a binary tree in which each node is labelled by the name of an inference rule plus an integer in the case of replacement rule (such that $\varphi \subseteq \sqsubseteq^{m}_{\Sigma,P}$). Then two proofs with the same sketch have the same complexity scheme.

§5 Proof of the theorem 5

First, we have to put any proof in some "standard form", which will be a proof using at most one application of the restriction rule, at the top of the associated tree.

Let $\|\!\vdash_{\Sigma}$ be the least binary relation over formulas which satisfies rules (1) to (4) ; we call a proof (which does not use the restriction rule) in this system a standard proof.

For any proof t :

- we denote by Alph(t) the union of Alph(P) U Alph(Σ) U Alph(Q) for all labels $P \vdash_{\Sigma} Q$ of t.
- if ρ is a renaming such that $supp(\rho) \cap Alph(t) \neq \emptyset$, we denote by ρ(t) the proof obtained from t by replacing at each node its label, say $P \vdash_{\Sigma} Q$, by $\rho \times \rho(P) \vdash_{\rho(\Sigma)} \rho x \rho(Q)$. Obviously, it is a proof of $\rho \times \rho(Q)$ from $\rho \times \rho(P)$ in $\rho(\Sigma)$ if t is a proof of P from Q in Σ (trivial induction on the size of t).

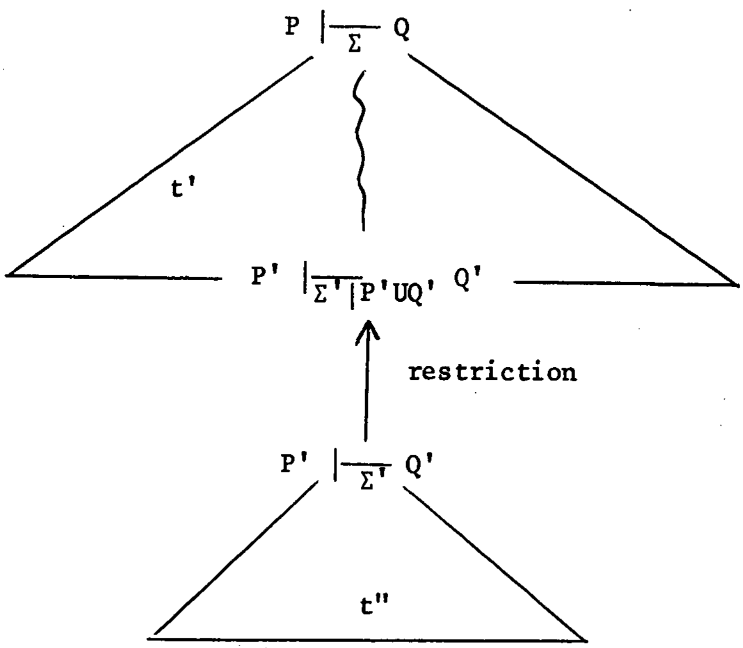
- if Σ is a RPS such that $Alph(\Sigma) \cap Alph(t) = \emptyset$, we get the tree $(t \leftarrow \Sigma)$ by replacing in t at all nodes the labels $P \vdash_{\Sigma'} Q$ by $P \vdash_{\Sigma' U \Sigma} Q$: obviously if t is a proof of P from Q in Σ', then $(t \leftarrow \Sigma)$ is a proof of P from Q in Σ' U Σ (same argument).

Lemma 1 : Standard form.

$$P \vdash_{\Sigma} Q \Rightarrow \exists \Sigma' : \Sigma' \mid P \cup Q = \Sigma \mid P \cup Q \text{ and } P \|\!\vdash_{\Sigma'} Q$$

We only sketch the proof (which proceeds by induction on the number of occurrences of the restriction rule in a proof of P from Q in Σ) :

Let t be a proof of the form :



and let ρ be a renaming such that, if we denote

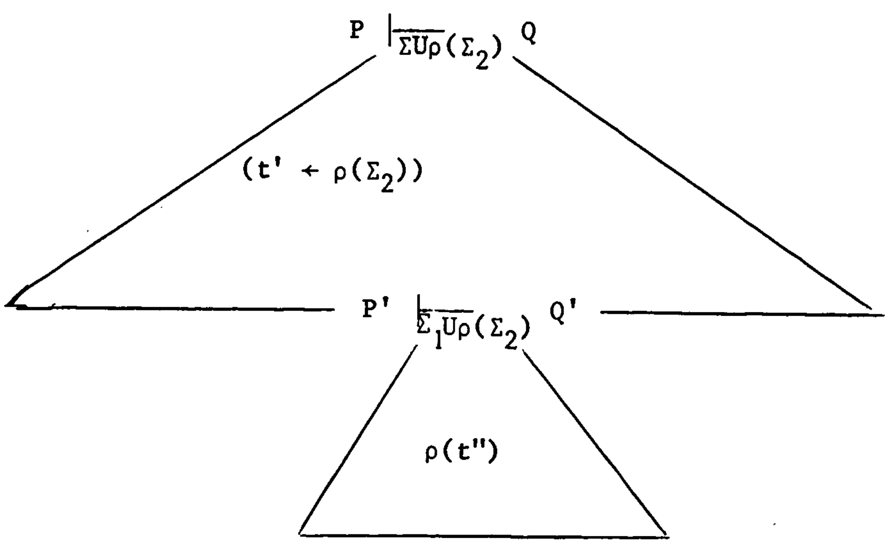
$$\Sigma_1 = \Sigma' \mid P' \cup Q' \text{ and } \Sigma_2 = \Sigma' - (P' \cup Q') :$$

$\mathrm{Alph}(t') \cap \mathrm{supp}(\rho) = \emptyset$ and $\mathrm{Alph}(\Sigma_2) \subseteq \mathrm{supp}(\rho)$

Then $\rho(\Sigma_1) = \Sigma_1$, $\rho \times \rho(P') = P'$ and $\rho \times \rho(Q') = Q'$.

Thus $\rho(t'')$ is a proof of Q' from P' in $\Sigma_1 \cup \rho(\Sigma_2) = \rho(\Sigma')$.

Since $\mathrm{Alph}(\rho(\Sigma_2)) \cap \mathrm{Alph}(t') = \emptyset$, we are able to construct the tree $(t' \leftarrow \rho(\Sigma_2))$ and to verify that



is again a proof of Q from P in $\Sigma_1 \cup \rho(\Sigma_2)$ and obviously $(\Sigma_1 \cup \rho(\Sigma_2)) \mid P \cup Q = \Sigma \mid P \cup Q$ since $Alph(\Sigma_2) \subseteq supp(\rho)$, and thus $Alph(\Sigma_2) \cap Alph(P \cup Q) = \emptyset$ (for $Alph(P \cup Q) \subseteq Alph(t'))$.

It is now clear that the theorem will be proved if it is established forstandard proofs since we have

$$P \subseteq I^S_\Sigma(\sigma, \{f\}) \Rightarrow P \subseteq I^S_{\Sigma|P}(\sigma, \{f\})$$

for $\Sigma|P = \Sigma|SUP$ ($S \subseteq M_A(V)^2$) and $\forall n,\ p \in N$:

$$\pi_\Sigma \circ \sigma^n_\Sigma \times \pi_\Sigma \circ \sigma^p(P) = \pi_{\Sigma|P} \circ \sigma^n_{\Sigma|P} \times \pi_{\Sigma|P} \circ \sigma^p_{\Sigma|P}(P)$$

From now on, we denote $\pi_\Sigma \circ \sigma^n_\Sigma$ by η^Σ_n.

Here is the second packet of technical definitions and properties :

- for a formula P, we define the parallel rewriting in P (see [18]) as :

$$t \underset{P}{\not\not\to} t' \Leftrightarrow_{def} \exists s \in M_{A\cup\Psi}(V)\ \exists\{x_{i_1},\dots,x_{i_p}\} \subseteq V$$

$$\exists \nu_1,\dots,\nu_p \text{ substitutions, } \exists\{t_j \subseteq t'_j / 1 \leqslant j \leqslant p\} \subseteq P \text{ s.t.}$$

$$t = s[\nu^*_j(t_j)/x_{i_j}]_{1\leqslant j\leqslant p} \text{ and}$$

$$t' = s[\nu^*_j(t'_j)/x_{i_j}]_{1\leqslant j\leqslant p}$$

Obviously, $\underset{P}{\to} \subseteq \underset{P}{\not\not\to} \subseteq \overset{*}{\underset{P}{\to}}$ and thus $\overset{*}{\underset{P}{\to}} = \overset{*}{\underset{P}{\not\not\to}}$

- for a RPS

$$\Sigma \begin{cases} \varphi_i x_1 \dots x_{n_i} = \tau_i \\ 1 \leqslant i \leqslant K \end{cases}$$

we define the simultaneous rewriting in Σ :

$t \underset{\Sigma}{\overset{\theta}{\to}} t' \Leftrightarrow_{def}$ (i) either $t = t'$

(ii) or $t = ft_1\dots t_k$, $t' = ft'_1\dots t'_k$, $f \in A\cup\Psi$ and $\forall i\ t'_i \underset{\Sigma}{\overset{\theta}{\to}} t'_i$

(iii) or $t = \varphi_i m_1 \dots m_{n_i}$, $t' = \tau_i[m'_1/x_1,\dots,m'_{n_1}/x_{n_1}]$ and $\forall j\ m_j \underset{\Sigma}{\overset{\theta}{\to}} m'_j$

Again $\underset{\Sigma}{\to} \subseteq \underset{\Sigma}{\overset{\theta}{\to}} \subseteq \overset{*}{\underset{\Sigma}{\to}}$ and $\overset{*}{\underset{\Sigma}{\to}} = \overset{*}{\underset{\Sigma}{\overset{\theta}{\to}}}$

We denote $\underset{\Sigma}{\xleftarrow{\theta}} = (\underset{\Sigma}{\xrightarrow{\theta}})^{-1}$

- As we shall see, only the $\underset{\Sigma}{\longleftarrow} \cup \underset{P}{\longrightarrow}$ steps (in an instance of the replacement rule) can increase the complexity of the formulas. Thus we define

$$\begin{cases} \sqsubseteq^{0}_{\Sigma,P} = (\sqsubseteq \cup \underset{\Sigma}{\xrightarrow{\theta}})^{*} \\ \\ \sqsubseteq^{m+1}_{\Sigma,P} = \sqsubseteq^{m}_{\Sigma,P} \circ (\underset{\Sigma}{\xleftarrow{\theta}} \cup \underset{P}{\not\!\!\!\longrightarrow}) \circ \sqsubseteq^{0}_{\Sigma,P} \end{cases}$$

It is clear that, if Q is (explicitely) a finite subset of $(\sqsubseteq \cup \underset{\Sigma}{\longleftrightarrow} \cup \underset{P}{\longrightarrow})^{*}$, then there exists a (computable) $m \in N$ such that $Q \subseteq \sqsubseteq^{m}_{\Sigma,P}$.

We left to the reader the (easy) verification of the following <u>facts</u>

(1) $t \sqsubseteq t' \Rightarrow \pi_{\Sigma}(t) \sqsubseteq \pi_{\Sigma}(t')$ and $\sigma_{\Sigma}(t) \sqsubseteq \sigma_{\Sigma}(t')$

(2) (see [28]) $t \underset{\Sigma}{\xrightarrow{\theta}} \sigma_{\Sigma}(t)$ and

$t \underset{\Sigma}{\xrightarrow{\theta}} t' \Rightarrow \pi_{\Sigma}(t) \sqsubseteq \pi_{\Sigma}(t'),\ \sigma_{\Sigma}(t) \underset{\Sigma}{\xrightarrow{\theta}} \sigma_{\Sigma}(t')$ and $t' \underset{\Sigma}{\xrightarrow{\theta}} \sigma_{\Sigma}(t)$

(3) $t \underset{P}{\not\!\!\!\longrightarrow} t' \Rightarrow \pi_{\Sigma}(t) \xrightarrow[\pi_{\Sigma} x \pi_{\Sigma}(P)]{} \pi_{\Sigma}(t')$ and, for all $n, p \in N$

$p \geq n \Rightarrow \sigma^{n}_{\Sigma}(t) (\xrightarrow[\sigma^{n}_{\Sigma} x \sigma^{P}_{\Sigma}(P)]{\not\;\;} \circ \underset{\Sigma}{\xrightarrow{p-n}}) \ \sigma^{p}_{\Sigma}(t')$

(hint : $\sigma_{\Sigma}(s\,[\nu^{*}_{j}(t_j)/x_{i_j}]_{1 \leq j \leq p}) = \sigma_{\Sigma}(s)[(\sigma_{\Sigma} \circ \nu_j)^{*}\ (\sigma_{\Sigma}(t_j))/x_{i_j}]_{1 \leq j \leq p}$)

(4) $\eta^{\Sigma}_{n} \times \eta^{\Sigma}_{q}(P) \subseteq \sqsubseteq_{S}$ and $p \geq q \Rightarrow \eta^{\Sigma}_{n} \times \eta^{\Sigma}_{p}(P) \subseteq \sqsubseteq_{S}$

(hint : recall that $\eta^{\Sigma}_{k} = \pi_{\Sigma} \circ \sigma^{k}_{\Sigma}$ - $p \geq q \Rightarrow$ by (2) $\eta^{\Sigma}_{q}(t) \sqsubseteq \eta^{\Sigma}_{p}(t)$)

<u>Lemma 2</u> :

Let t be a standard proof of Q from P in Σ, and Σ' independant from Σ. If we get the tree $t^{\Sigma'}_{m}$ (for m in N) by replacing at all nodes of t the labels $P' \mid\!\vdash_{\Sigma''} Q'$ by $\eta^{\Sigma'}_{m} \times \eta^{\Sigma'}_{m}(P') \mid\!\vdash_{\Sigma'\Sigma''} \eta^{\Sigma'}_{m} \times \eta^{\Sigma'}_{m}(Q')$, then $t^{\Sigma'}_{m}$ is a standard proof of $\eta^{\Sigma'}_{m} \times \eta^{\Sigma'}_{m}(Q)$ from $\eta^{\Sigma'}_{m} \times \eta^{\Sigma'}_{m}(P)$ in Σ.

<u>Proof</u> : by induction on t.

(1) If t reduces to a single node labelled $P \mid\vdash_{\Sigma} Q$, which is an instance of the replacement rule, then there exists $k \in N$ such that $Q \subseteq \sqsubseteq^{k}_{\Sigma,P}$.
Let us prove the lemma in this case by induction on k ; and in fact : $\eta^{\Sigma'}_{m} \times \eta^{\Sigma'}_{m}(Q) \subseteq \sqsubseteq^{k}_{\Sigma,\eta^{\Sigma'}_{m}\times\eta^{\Sigma'}_{m}(P)}$:

(i) if $k = 0$, $Q \subseteq (\sqsubseteq \cup \xrightarrow[\Sigma]{\theta})^{*}$. We have only to check (by means of fact (1)) that

$$t \xrightarrow[\Sigma]{\theta} t' \Rightarrow \forall m \in N \quad \eta^{\Sigma'}_{m}(t) \xrightarrow[\Sigma]{\theta} \eta^{\Sigma'}_{m}(t')$$

which is true because Σ and Σ' are independant (trivial induction on the definition of $\xrightarrow[\Sigma]{\theta}$).

(ii) if $Q \subseteq \sqsubseteq^{k+1}_{\Sigma,P}$ then for all $(t,t') \in Q$ there exists s,s' s.t. :

$$t \sqsubseteq^{k}_{\Sigma,P} s(\xleftarrow[\Sigma]{\theta} \cup \xrightarrow[P]{/\!/}) s'(\sqsubseteq \cup \xrightarrow[\Sigma]{\theta})^{*} t'$$

By induction hypothesis : $\eta^{\Sigma'}_{m}(t) \sqsubseteq^{k}_{\Sigma,\eta^{\Sigma'}_{m}\times\eta^{\Sigma'}_{m}(P)} \eta^{\Sigma'}_{m}(s)$.

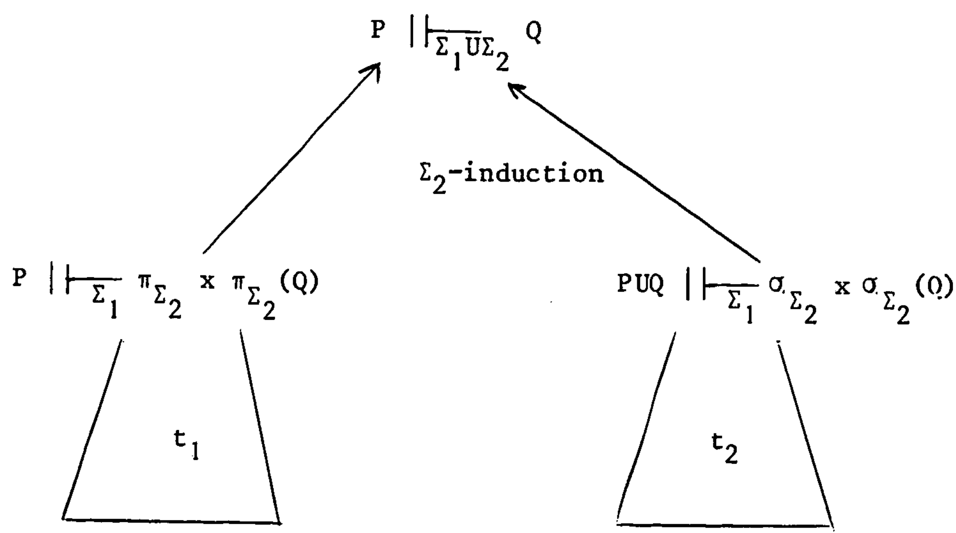
(iii) if $s \xleftarrow[\Sigma]{\theta} s'$ then, as in case (i) $\eta^{\Sigma'}_{m}(s) \xleftarrow[\Sigma]{\theta} \eta^{\Sigma'}_{m}(s')$ and, as we have seen in (i) : $\eta^{\Sigma'}_{m}(s')(\sqsubseteq \cup \xrightarrow[\Sigma]{\theta})^{*} \eta^{\Sigma'}_{m}(t')$

(iv) if $s \xrightarrow[P]{/\!/} s'$ then by (3) $\eta^{\Sigma'}_{m}(s) \xrightarrow[\eta^{\Sigma'}_{m}\times\eta^{\Sigma'}_{m}(P)]{/\!/} \eta^{\Sigma'}_{m}(s')$ and again

$$\eta^{\Sigma'}_{m}(s')(\sqsubseteq \cup \xrightarrow[\Sigma]{\theta})^{*} \eta^{\Sigma'}_{m}(t').$$

(2),(3) If t is a tree in which the top rule is the union or cut rule, then the lemma is trivial for $t^{\Sigma'}_{m}$.

(4) If t is a tree (where $\Sigma_1 \cup \Sigma_2 = \Sigma$)



then we apply the induction hypothesis on t_1 and t_2, and only remark that

Σ and Σ' independant $\Rightarrow \forall m \in N$

$$\eta_m^{\Sigma'} \times \eta_m^{\Sigma'}(\pi_{\Sigma_2} \times \pi_{\Sigma_2}(Q)) = \pi_{\Sigma_2} \times \pi_{\Sigma_2}(\eta_m^{\Sigma'} \times \eta_m^{\Sigma'}(Q)) \text{ and}$$

$$\eta_m^{\Sigma'} \times \eta_m^{\Sigma'}(\sigma_{\Sigma_2} \times \sigma_{\Sigma_2}(Q)) = \sigma_{\Sigma_2} \times \sigma_{\Sigma_2}(\eta_m^{\Sigma'} \times \eta_m^{\Sigma'}(Q)).$$

The crucial step in the proof of the theorem is the following claim.

(1) for any (explicit) standard proof t of Q from P in Σ there (effectively) exists a functionnal of $\mathcal{O}$, denoted ψ and called the complexity scheme of t, such that

$\forall f \in \mathcal{F} : P \subseteq I_\Sigma^S(\sigma,\{f\}) \Rightarrow Q \subseteq I_\Sigma^S(\sigma,\{\psi(f)\})$

(2) moreover, if Σ' is independant from Σ, then the same functionnal ψ of $\mathcal{O}$ is the complexity scheme of $t_m^{\Sigma'}$ (as defined in lemma 2), for all m in N.

The theorem follows from this claim, since $S \subseteq M_A(V)^2$ implies, for all Σ, $S \subseteq I_\Sigma^S(\sigma,\{id\})$.

Proof of the claim : by induction on a standard proof t of Q from P in Σ :

(1) if t reduces to one node, labelled $P \mid\mid\!\!\frac{\quad}{\Sigma} Q$, then this is an (explicit) instance of the replacement rule, thus there (effectively) exists k of N such that $Q \subseteq \sqsubseteq^k_{\Sigma,P}$. Let us define

$$\begin{cases} \psi_0 = \underline{id} \\ \psi_{m+1} = \underline{comp}(\underline{suc}, \psi_m) \end{cases}$$

or more concretely, for $f \in N^N$:

$$\begin{cases} \psi_0(f)(n) = f(n) \\ \psi_{m+1}(f)(n) = f(\psi_m(f)(n)) + 1 \end{cases}$$

We prove that $\forall m \in N : \psi_m \in \mathcal{O}$ (which is trivial) and $\forall f \in \mathcal{F}$:

$P \subseteq I^S_\Sigma(\sigma,\{f\}) \Rightarrow Q \subseteq I^S_\Sigma(\sigma,\{\psi_k(f)\})$

by induction on k :

(i) if $k = 0$ then for all $(t,t') \in Q$ we have $t(\sqsubseteq \cup \xrightarrow[\Sigma]{\theta})^* t'$ and (by facts (1) & (2)) :

$\forall n \in N \quad \eta^\Sigma_n(t) \sqsubseteq \eta^\Sigma_n(t')$

But $\psi_0(f)(n) = f(n)$ and $f \in \mathcal{F} \Rightarrow f(n) \geq n$ thus (by fact (2))

$\forall n \in N \quad \eta^\Sigma_n(t) \sqsubseteq \eta^\Sigma_{\psi_0(f)(n)}(t')$

(ii) if $Q \subseteq \sqsubseteq^{k+1}_{\Sigma,P}$ then, for all (t,t') there exists s,s' s.t. :

$t \sqsubseteq^k_{\Sigma,P} s(\xleftarrow[\Sigma]{\theta} \cup \xrightarrow[P]{/\!/}) s'(\sqsubseteq \cup \xrightarrow[\Sigma]{\theta})^* t'$

By induction hypothesis :

$\forall n \in N : \quad \eta^\Sigma_n(t) \sqsubseteq_S \quad \eta^\Sigma_{\psi_k(f)(n)}(s)$

(iii) if $s \xleftarrow[\Sigma]{\theta} s'$ then (by fact (2)) $s \xrightarrow[\Sigma]{\theta} \sigma_\Sigma(s')$ and (fact (2))

$\forall p \in N : \eta^\Sigma_p(s) \sqsubseteq \eta^\Sigma_{p+1}(s')$

But $f \in \mathcal{F} \Rightarrow f(\psi_k(f)(n)) + 1 \geq \psi_k(f)(n) + 1$ (since $\forall q\ f(q) \geq q$) and by definition of ψ_{k+1} and fact (2) :

$\forall n \in \eta_{\psi_k(f)(n)}(s) \sqsubseteq \eta_{\psi_{k+1}(f)(n)}(s')$

and, as we have seen in (i) :

$\eta^\Sigma_{\psi_{k+1}(f)(n)}(s') \sqsubseteq \eta^\Sigma_{\psi_{k+1}(f)(n)}(t')$

(iv) if $s \xrightarrow[P]{/\!/} s'$ then by facts (3) & (2) :

$\eta^\Sigma_{\psi_k(f)(n)}(s) \ (\xrightarrow[P']{/\!/} \circ \sqsupseteq) \ \eta^\Sigma_{\psi_{k+1}(f)(n)}(s')$

where $P' = \eta^\Sigma_{\psi_k(f)(n)} \times \eta^\Sigma_{\psi_{k+1}(f)(n)}(P)$

since $\psi_{k+1}(f)(n) \geq \psi_k(f)(n)$ (see (iii)).

But by hypothesis $\eta^\Sigma_{\psi_k(f)(n)} \times \eta^\Sigma_{f(\psi_k(f)(n))}(P) \subseteq \sqsubseteq_S$

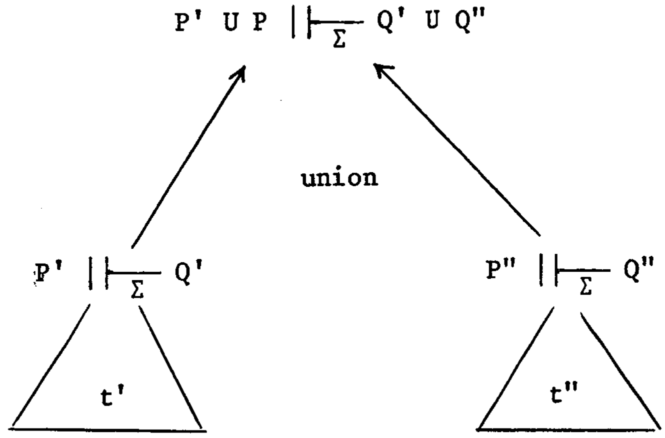
thus by fact (4) : $\xrightarrow[P]{/\!/} \subseteq \sqsubseteq_S$ since $\psi_{k+1}(f)(n) \geq f(\psi_k(f)(n))$.

Again, as we have seen in (i) :

$$\eta^{\Sigma}_{\psi_{k+1}(f)(n)}(s') \sqsubseteq \eta^{\Sigma}_{\psi_{k+1}(f)(n)}(t')$$

To achieve the proof of the claim in this case, we only recall the proof of lemma 2, in which it is shown that we have $\eta^{\Sigma'}_{m} \times \eta^{\Sigma'}_{m}(Q) \subseteq \sqsubseteq^{k}_{\Sigma, \eta^{\Sigma'}_{m} \times \eta^{\Sigma'}_{m}}(P)$ for all m, and the "complexity scheme" ψ of t only depends on k in this case.

(2) Assume that t is the tree



and that ψ' and ψ'' are the complexity schemes for t' and t", and define $\psi = \underline{\max}(\psi', \psi'')$, obviously we have

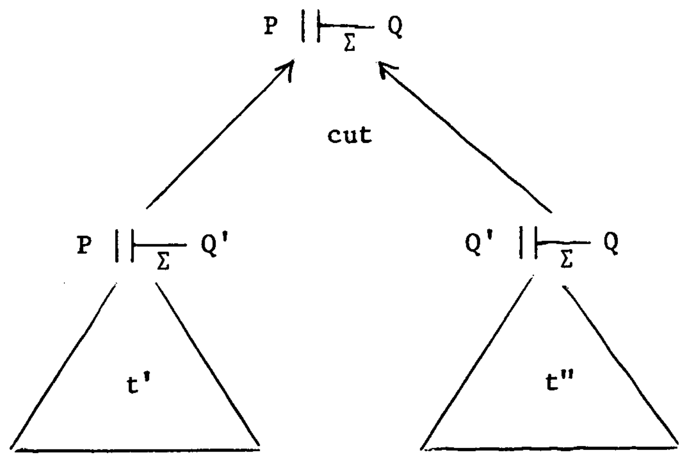
$$\forall f \in \mathfrak{F} : P' \cup P'' \subseteq I^{S}_{\Sigma}(\sigma, \{f\}) \Rightarrow Q' \cup Q'' \subseteq I^{S}_{\Sigma}(\sigma, \{\psi(f)\})$$

since $R \subseteq I^{S}_{\Sigma}(\sigma, \{g\})$ and $g \leqslant h$ ($\Leftrightarrow \forall n : g(n) \leqslant h(n)$)

implies $R \subseteq I^{S}_{\Sigma}(\sigma, \{h\})$.

The second point of the claim is also easy in this case.

(3) If t is the tree

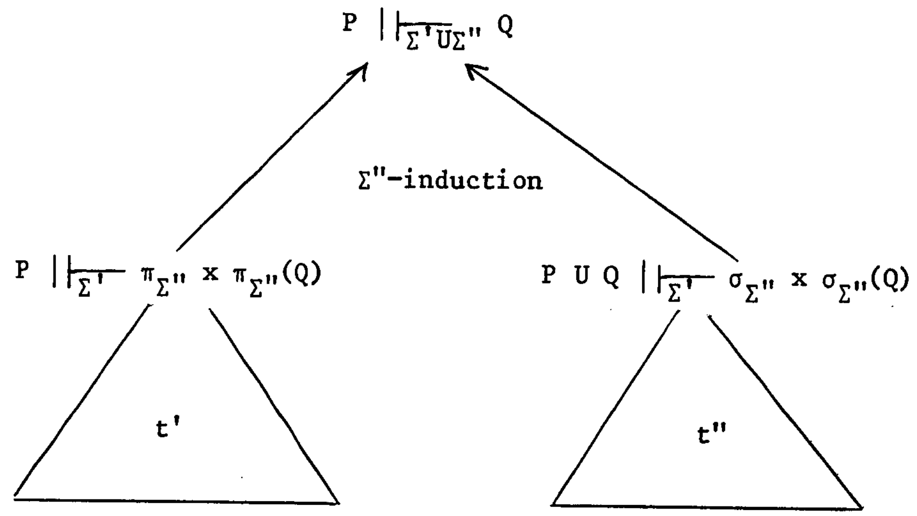


and if ψ' and ψ'' are the complexity schemes for t' and t'' , then $\psi = \psi'' \circ \psi'$ is the complexity scheme of t since

$\forall f \in \mathcal{S} : P \subseteq I^S_\Sigma(\sigma,\{f\}) \Rightarrow Q' \subseteq I^S_\Sigma(\sigma,\{\psi'(f)\}) \Rightarrow Q \subseteq I^S_\Sigma(\sigma,\{\psi''(\psi'(f))\})$.

Again, the second point of the claim is trivial here.

(4) If t is the tree



Let ψ' be (given by the induction hypothesis) a scheme of complexity for t' :

$\forall f \in \mathcal{S} \quad P \subseteq I^S_{\Sigma'}(\sigma,\{f\}) \Rightarrow \pi_{\Sigma''} \times \pi_{\Sigma''}(Q) \subseteq I^S_{\Sigma'}(\sigma,\{\psi'(f)\})$

First, let us remark that, since P is independant of Σ'' :

$\forall p, q \in N \quad \eta_q^{\Sigma''} \times \eta_p^{\Sigma''}(P) = P$ and thus

$P \subseteq I^S_{\Sigma'}(\sigma,\{f\}) \Leftrightarrow P \subseteq I^S_{\Sigma' \cup \Sigma''}(\sigma,\{f\})$

since $\forall n \in N : \eta_n^{\Sigma'} \circ \eta_n^{\Sigma''} = \eta_n^{\Sigma''} \circ \eta_n^{\Sigma'} = \eta_n^{\Sigma' \cup \Sigma''}$

Let ψ'' be (given by the induction hypothesis) a scheme of complexity for t'', and define a sequence of elements of $\mathcal{O}$ by

$$\begin{cases} \lambda(0) = \psi' \\ \lambda(m+1) = \psi'' \circ \lambda(m) \end{cases}$$

We prove that

$P \subseteq I^S_{\Sigma'}(\sigma,\{f\}) \Rightarrow \forall m \in N \quad \eta_m^{\Sigma''} \times \eta_m^{\Sigma''}(Q) \subseteq I^S_{\Sigma'}(\sigma,\{\lambda(m)(f)\})$.

(i) if $m = 0$, then $\eta_m^{\Sigma''} \times \eta_m^{\Sigma''}(Q) = \pi_{\Sigma''} \times \pi_{\Sigma''}(Q)$ and $\mathcal{A}(m) = \psi'$, this is nothing else that the induction hypothesis on t'.

(ii) We have $\eta_{m+1}^{\Sigma''} \times \eta_{m+1}^{\Sigma''}(Q) = \eta_m^{\Sigma''} \times \eta_m^{\Sigma''}(\sigma_{\Sigma''} \times \sigma_{\Sigma''}(Q))$, by definition of $\eta_p^{\Sigma''}$. Since Σ' and Σ'' are independant, we can use lemma 2, which indicates that $t''^{\Sigma''}_m$ is a proof of $\eta_{m+1}^{\Sigma''} \times \eta_{m+1}^{\Sigma''}(Q)$ from $\eta_m^{\Sigma''} \times \eta_m^{\Sigma''}(P \cup Q)$ in Σ', with the same complexity scheme ψ'' (induction hypothesis of the proof of the claim, point (2)). But

$\eta_m^{\Sigma''} \times \eta_m^{\Sigma''}(P \cup Q) = P \cup \eta_m^{\Sigma''} \times \eta_m^{\Sigma''}(Q)$ (independance of P and Σ'')

and by the induction hypothesis on m :

$\eta_m^{\Sigma''} \times \eta_m^{\Sigma''}(Q) \subseteq I_{\Sigma'}^S(\sigma, \{\mathcal{A}(m)(f)\})$ if $P \subseteq I_{\Sigma'}^S(\sigma, \{f\})$

Obviously $P \subseteq I_{\Sigma'}^S(\sigma, \{\mathcal{A}(m)(f)\})$ in this case since $\mathcal{A}(m)(f) \geq f$ (for all $\psi \in \mathcal{O}$ and $f \in \mathcal{F}$: $\psi(f) \geq f$).

Thus $\eta_m^{\Sigma''} \times \eta_m^{\Sigma''}(P \cup Q) \subseteq I_{\Sigma'}^S(\sigma, \{\mathcal{A}(m)(f)\})$ which implies, by definition of $\mathcal{A}$ $(\mathcal{A}(m+1)(f) = \psi''(\mathcal{A}(m)(f)))$

$$\eta_{m+1}^{\Sigma''} \times \eta_{m+1}^{\Sigma''}(Q) \subseteq I_{\Sigma'}^S(\sigma, \{\mathcal{A}(m+1)(f)\}) -$$

This means that, if $P \subseteq I_{\Sigma'}^S(\sigma, \{f\})$ then $\forall m,n \in N$:

$$\eta_m^{\Sigma''} \times \eta_m^{\Sigma''}(\eta_n^{\Sigma'} \times \eta_{(\mathcal{A}(m)(f))(n)}^{\Sigma'}(Q)) \subseteq \sqsubseteq_S$$

Since $\mathcal{A}(m)(f)(n) \geq n$ for all n, if we let $m = n$ in this formula, we have by fact (4) : for all $n \in N$

$$\eta_n^{\Sigma'' \cup \Sigma'} \times \eta_{\psi(f)(n)}^{\Sigma'' \cup \Sigma'}(Q) \subseteq \sqsubseteq_S$$

where $\psi(f)(n) = (\mathcal{A}(n)(f))(n)$. But ψ is nothing but $\delta(\psi', \psi'')$, which is thus a complexity scheme for t.

The verification of the second point of the claim in this case is trivial (see lemma 2).

V. References.

1. ADJ "Initial algebra semantics and continuous semantics", J. of Assoc. Comput. Mach., 24, 1 (1977) 68-95.

2. de Bakker J., "Recursive procedures", Math. Center, Amsterdam (1971).

3. de Bakker J., Scott D., "A theory of programs", IBM Seminar, unpublished manuscript, Vienna (1971).

4. Boudol G., "A new recursion induction principle", 4th GI, LNCS 67, Springer-Verlag ed., Aachen (1979) 79-90.

5. Burstall R., "Proving properties of programs by structural induction", Comput. J., 12 (1969) 41-48.

6. Burstall R., Darlington J., "A transformation system for developping recursive programs", J. Assoc. Comput. Mach., 24, 1 (1977) 46-67.

7. Cadiou J.-M., "Recursive definition of partial functions and their computations", Ph. D. Thesis, Stanford (1972).

8. Courcelle B., "A representation of trees by languages", Theor. Comput. Sci. 6, 3, 255-279 & 7, 1, 25-55 (1978).

9. Courcelle B., "Infinite trees in normal form and recursive equations having a unique solution", Math. System Theory, 13 (1979) 131-180.

10. Courcelle B., Guessarian I., "On some classes of interpretations", J. of Comput. System Sci., 17 (1978), 388-413.

11. Courcelle B., Nivat M., "The algebraic semantics of recursive program schemes", MFCS 78, LNCS 64, Springer-Verlag ed. (1978) 16-30.

12. Courcelle B., Vuillemin J., "Completeness results of the equivalence of recursive program schemes", J. of Comput. System Sci. (1976) 179-197.

13. Downey P., Sethi R., "Correct computation rules for recursive languages", SIAM J. of Computing, 5 (1976).

14. Greif I., "Induction in proofs about programs", M. Sc. Thesis, M.I.T., Cambridge, Mass. (1972).

15. Guessarian I., "Equivalences sémantiques et familles d'interprétations", Doctoral Thesis, Paris (1975).

16. Guessarian I., "Semantic equivalence of program schemes and its syntactic characterization", 3rd ICALP, Edinburg (1976) 189-200.

17. Harrison M., Havel I., "Strict deterministic grammars", J. of Comput. System Sci., 7 (1973) 237-277.

18. Huet G., "Confluent reductions : abstract properties and applications to term rewriting systems", 18th FOCS (1977) 30-45.

19. Huet G., Hullot J.-M., "Proofs by induction in equationnal theories with constructors", preliminary paper (1980).

20. Huet G., Lang B., "Proving and applying program transformations expressed with 2nd order patterns", Acta Infor., 11 (1978) 31-55.

21. Huet G., Oppen D., "Equations and rewrite rules. A survey", preliminary paper (1979).

22. Kott L., "About transformation system : a theoretical study", in "Program Transformations" (B. Robinet ed.), Dunod, Paris (1978) 232-247.

23. Kott L., "Des substitutions dans les systèmes d'équations algébriques. Application aux transformations de programme et à leur correction", Doctoral Thesis, Paris (1979).

24. Kott L., "A system for proving equivalence of recursive programs", Proc. of 5th Coll. on Automated Deduction, les Arcs (1980).

25. Mac Carthy J., "A basis for a mathematical theory of computation" in "Computer Programming and Formal Systems" (P. Braffort, D. Hirschberg eds.), North-Holland, Amsterdam (1963).

26. Manna Z., "Mathematical theory of computation", Mc Graw-Hill, New-York (1974).

27. Morris J., "Another recursion induction principle", Comm. of Assoc. Comput. Mach., 14, 5 (1971) 351-354.

28. Nivat M., "On the interpretation of recursive polyadic program scheme", Instituto Nazionale di Alta Matematica, Symposia Matematica, Vol. XV (1975) 255-281.

29. Nivat M., "Interprétation universelle d'un schéma de programme récursif", Informatica, 7, 1 (1977) 9-16.

30. Park D., "Fixpoint induction and proofs of program properties" in Machine Intel. 5 (B. Meltzler, D. Michie eds.), Edinburg Univ. Press (1970) 59-78.

31. Rosen B., "Program Equivalence and Context-free Grammars", J. of Comput. Science Sci., 11 (1976) 358-374.

32. Rounds W., "Tree oriented proofs of some theorems on context-free languages and indexed languages", 2nd STOC (1970) 109-116.

33. Schützenberger M.-P., "Pushdown automata and context-free languages", Infor. and Control, 6 (1963) 246-264.

34. Vuillemin J., "Correct and optimal implementation of recursion in a simple programming language", J. of Comput. System Sci., 9 (1974) 332-354.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique